MCR-81-576

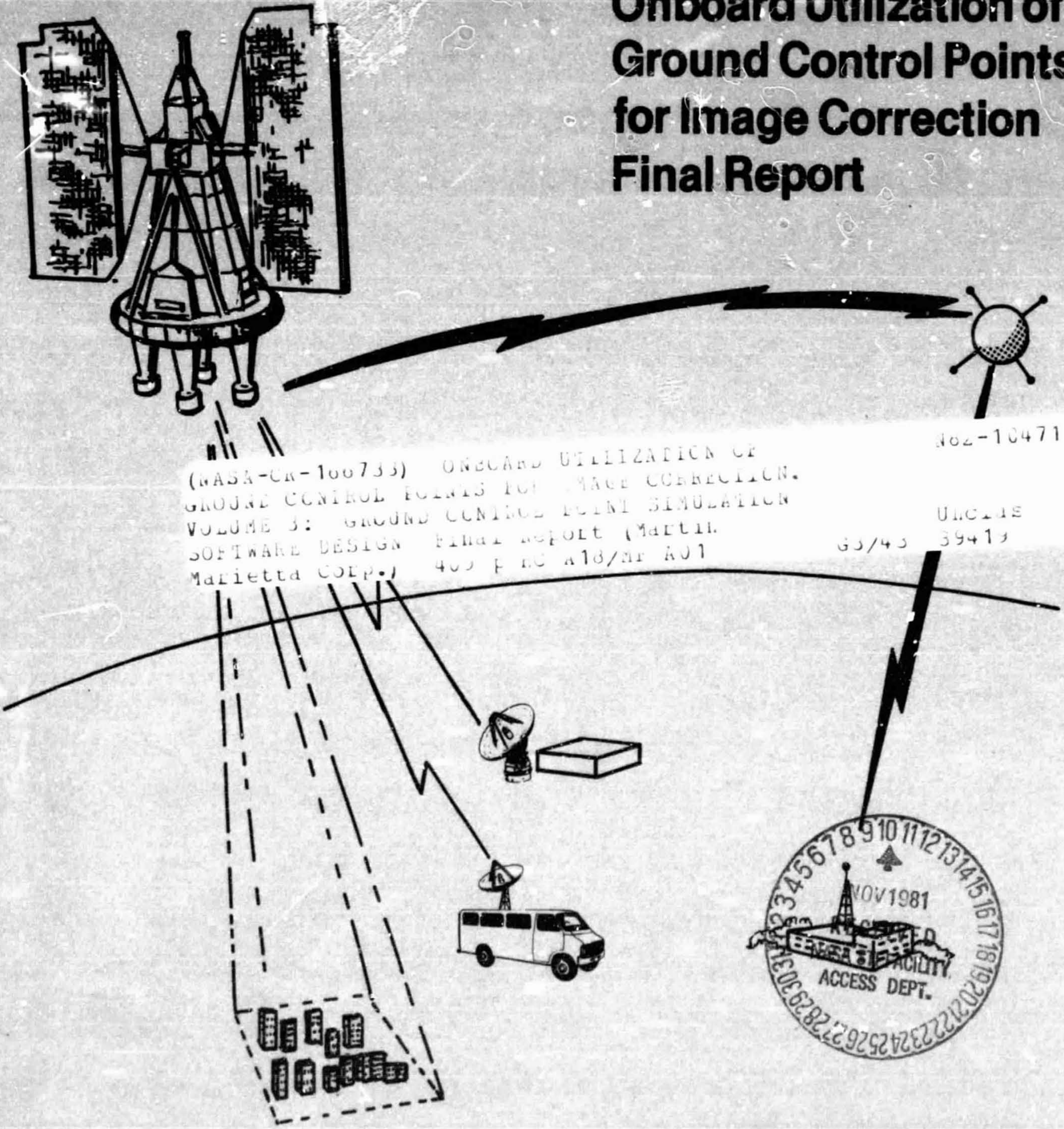
NASA CR-166733

Volume III

Ground Control
Point Simulation
Software Design

May 1981

Onboard Utilization of Ground Control Points for Image Correction
Final Report



(NASA-CR-166733) ONBOARD UTILIZATION OF GROUND CONTROL POINTS FOR IMAGE CORRECTION. VOLUME 3: GROUND CONTROL POINT SIMULATION SOFTWARE DESIGN Final Report (Martin Marietta Corp.) 409 p HC A18/MF A01

N82-10471

Unclas
G3/43 39413

MARTIN MARIETTA

Volume III

Ground Control
Point Simulation
Software Design

May 1981

ONBOARD UTILIZATION OF GROUND CONTROL POINTS FOR IMAGE CORRECTION FINAL REPORT

MARTIN MARIETTA
DENVER AEROSPACE
Denver, Colorado 80201

V. FOREWORD

This volume of the "Onboard Utilization of Ground Control Points for Image Correction" final report provides a detailed description of the software that was developed to simulate the ground control point navigation system. Three other volumes have been incorporated into the final report. Volume I provides an executive summary, Volume II contains a detailed description of the study and simulator results, and Volume IV is an appendix describing software utilized for image correction accuracy measurement.

TABLE OF CONTENTS

Page

1.0 INTRODUCTION

The Ground Control Point Simulation Program (GCPSIM) has been designed as an analysis tool to predict the performance of the navigation system illustrated in Figure A-1. The system consists of two star trackers, a GPS receiver, a gyro package, and a landmark tracker.

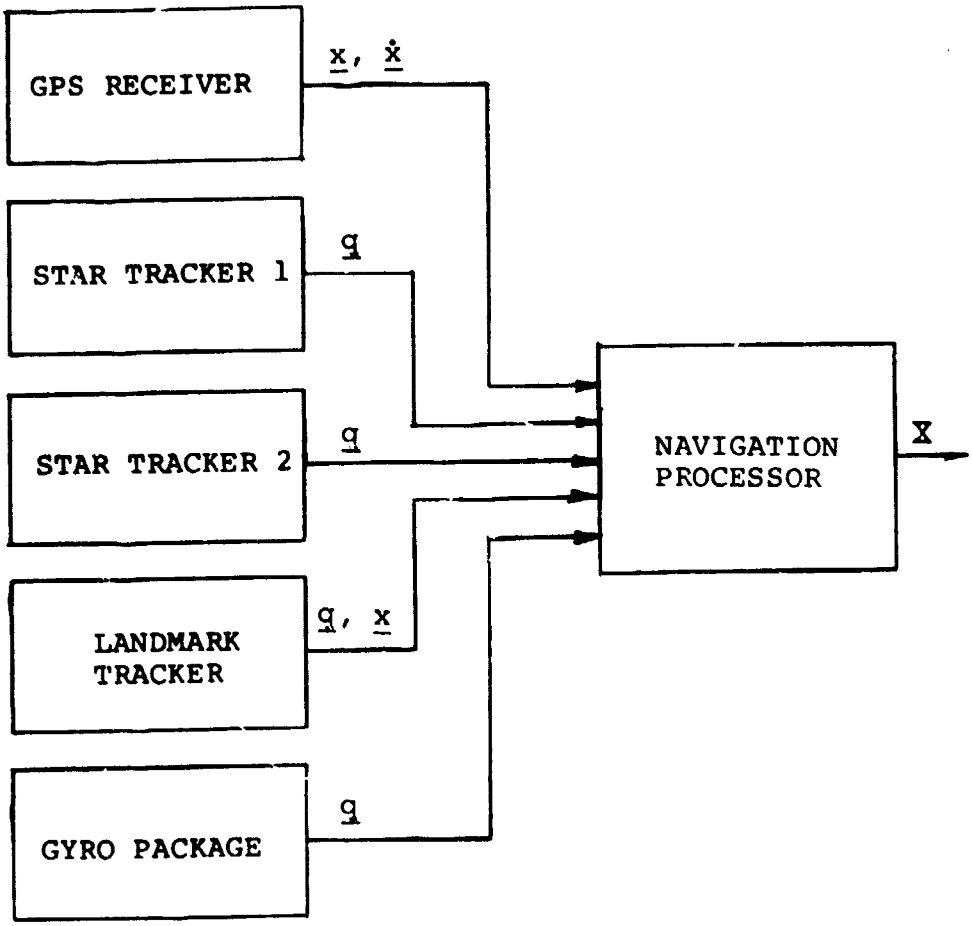


Figure A-1 System Configuration

The program has been configured to provide an extensive error analysis rather than simply a covariance analysis. A covariance analysis provides a great deal of information but in many cases this information is inaccurate and misleading. For example, there are cases where the covariance matrix converges over a period of time while the actual state diverges. These types of instability are common in Kalmar filters and without direct knowledge of the state error conclusions can be inaccurate. For this reason, GCPSIM has been designed to model the true state of the vehicle using the dynamic equations of motion. By propagating the true state, it is not only possible to perform a covariance analysis, but to generate the actual state error and measurement residuals as well.

GCPSIM has been designed to accept either simulated GCP measurements or measurements that have been extracted directly from corrected Landsat imagery. These capabilities are included to allow analysis over long periods of time as well as to demonstrate the feasibility.

REPORT FORMAT

The intent of this document is to give visibility to the GCPSIM code. GCPSIM uses top-down structured programming aided by visual control logic representation (VCLR). A total system hierarchy diagram aids in depicting the relationship between the modules.

Each module is presented separately with a general discussion of the function performed followed by requirements, algorithm and process, VCLR, and source code (Fortran IV). The remainder of this section will describe the VCLR presentation aid.

Visual Control Logic Representation (VCLR)

VCLR charts give control logic which is compatible with structured programming. It offers many advantages and flowcharting.

1) Only the standard constructs are used.

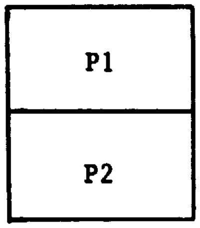
2) The total scope and impact of the logic can be seen and easily understood.

3) No extraneous symbols, connections, or notations are used.

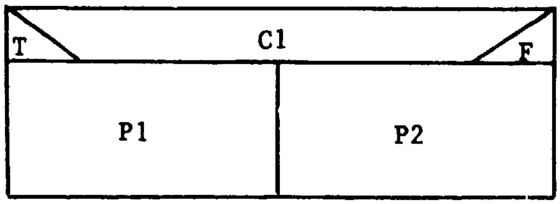
VCLR provides visible control logic representation which is a picture of a software design. Its primary purpose is to enable software engineers to express their thinking visually. Concentration is on the control logic of the design.

Standard constructs in visible control logic representations are the same as those for pseudo-code: SEQUENCE, IFTHENELSE, DOWHILE, DOUNTIL and DOCASE. Only the representations differ.

SEQUENCE - A SEQUENCE is simply one standard construct or one single statement followed by another. If "P1" and "P2" are standard constructs or single statements, the sequence would appear in a visible control logic representation as:



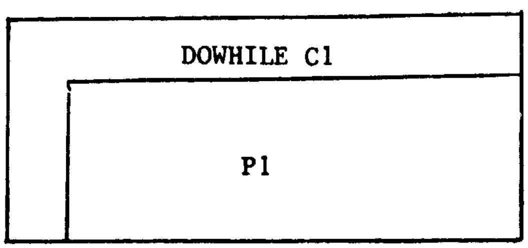
IFTHENELSE - Consists of a true/false test, and a path for each state. The true path appears on the left side, under the "T". One of the paths may be a "do-nothing" or "NULL" path. One or both paths must consist of a standard construct or of a single statement. If "C1" is the condition being tested, "P1" is the true path and "P2" is the false path, the IFTHENELSE construct would be written as



DOWHILE - The DOWHILE is a loop with these characteristics:

1) The counter or other item to be "incremented" is initialized before entering the loop.
2) The test is performed at the beginning of the loop. The conditions which must exist in order for the loop top be executed are the conditions which appear in the DOWHILE test.
3) The item to be executed must be a standard construct or a single statement.
4) The counter is incremented or other increment-like action is generally taken (e.g., another line is read) at the end of the loop.

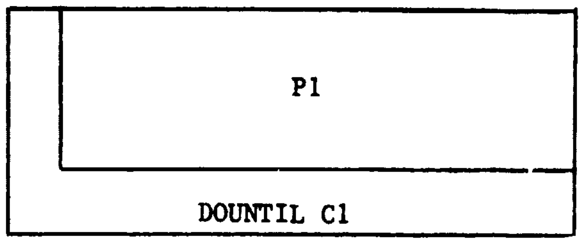
If "C1" is the condition which must exist for the loop to be executed, and "P1" is a standard construct or single statement, the DOWHILE would be written as follows:



DOUNTIL - The DOUNTIL is a loop with these characteristics:

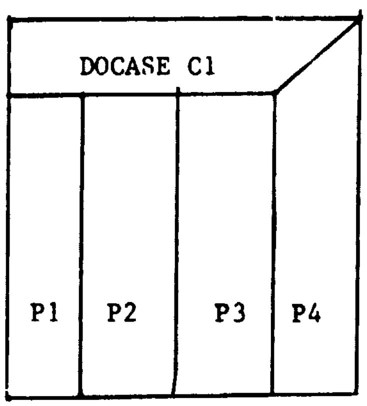
1) The counter or other item to be "incremented" is initialized before entering the loop.
2) The test is performed at the end of the loop. The conditions which must exist in order to exit from the loops are those which appear in the DOUNTIL test.
3) The item to be executed must be a standard construct or a single statement.
4) The counter is incremented or other increment-like action is generally taken (e.g., another line is read) at the beginning of the loop.

If "C1" is the condition which must exist to exit from the loop, and "P1": is a standard construct or single statement, the DOUNTIL would be written as follows:

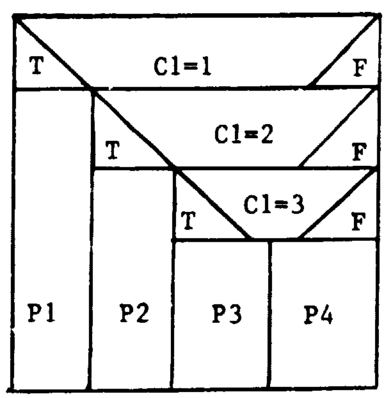


DOCASE - Use the DOCASE construct when you want to execute a different set of statements for each of several different values of a variable.

If "C1" is the variable being tested, and if "C1" may have values of 1, 2 or 3, the construct appears as follows:



Example A



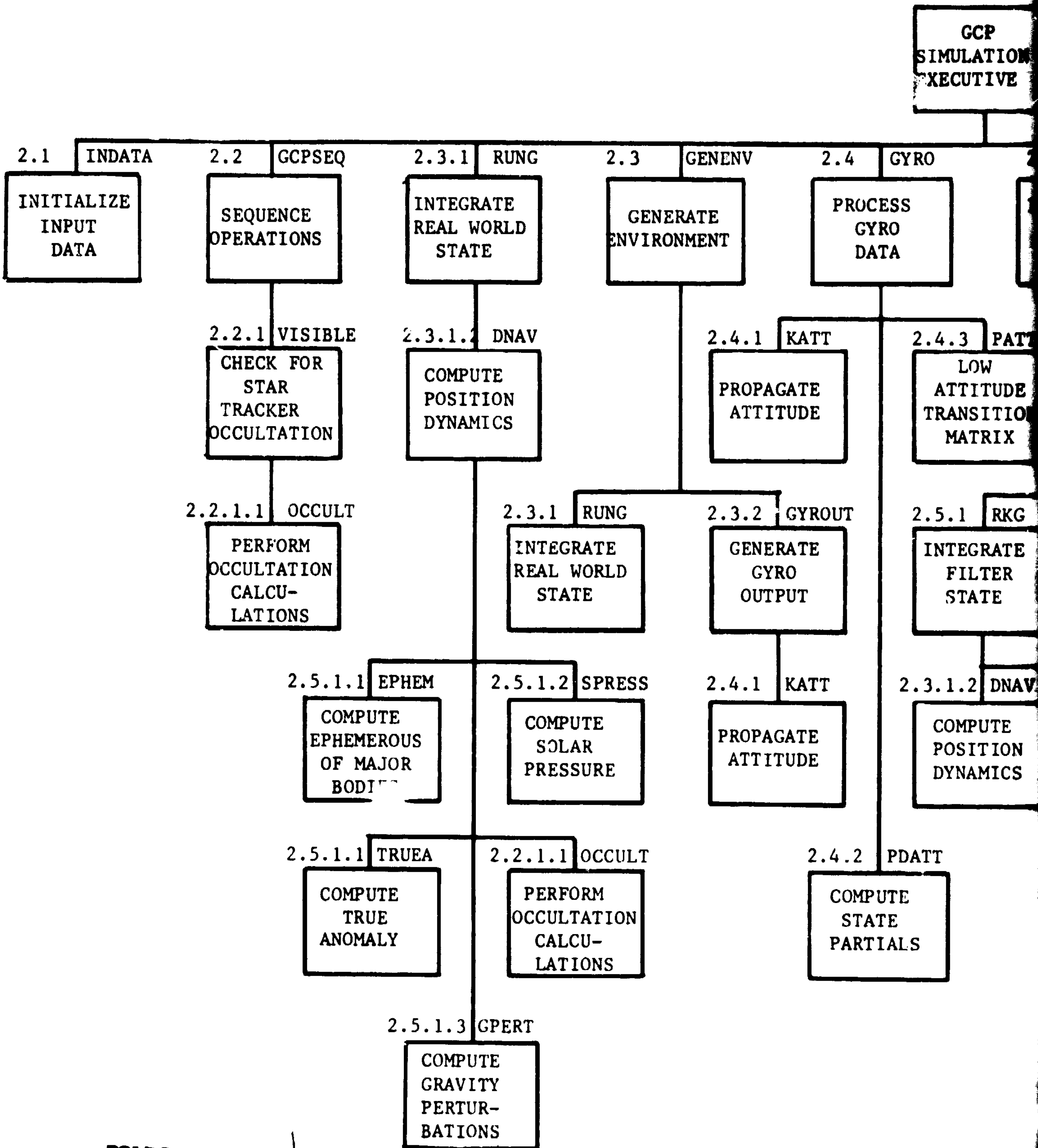
Example B

Example "A" is equivalent to the nested IFTHENELSE form shown in "B".

TOTAL SYSTEM HIERARCHY DIAGRAM

The total system hierarchy diagram (Figure A-2) is designed to illustrate the functional structure of GCPSIM. The upper most block is the main program responsible for simulation control. The eleven blocks imediately below this type block represent the subroutines called by the main program. These programs in turn call subroutines or the next level down and so forth.

This report has been separated according to a functional decomposition of the program. Each of the eleven main subroutines are described in separate chapters along with all the subsequent tasks with which they interact. For example, the chapter describing the generate measurement module, MEASURE, is broken into three main sections corresponding to each of the measurement types generated.



FOLDOUT FRAME

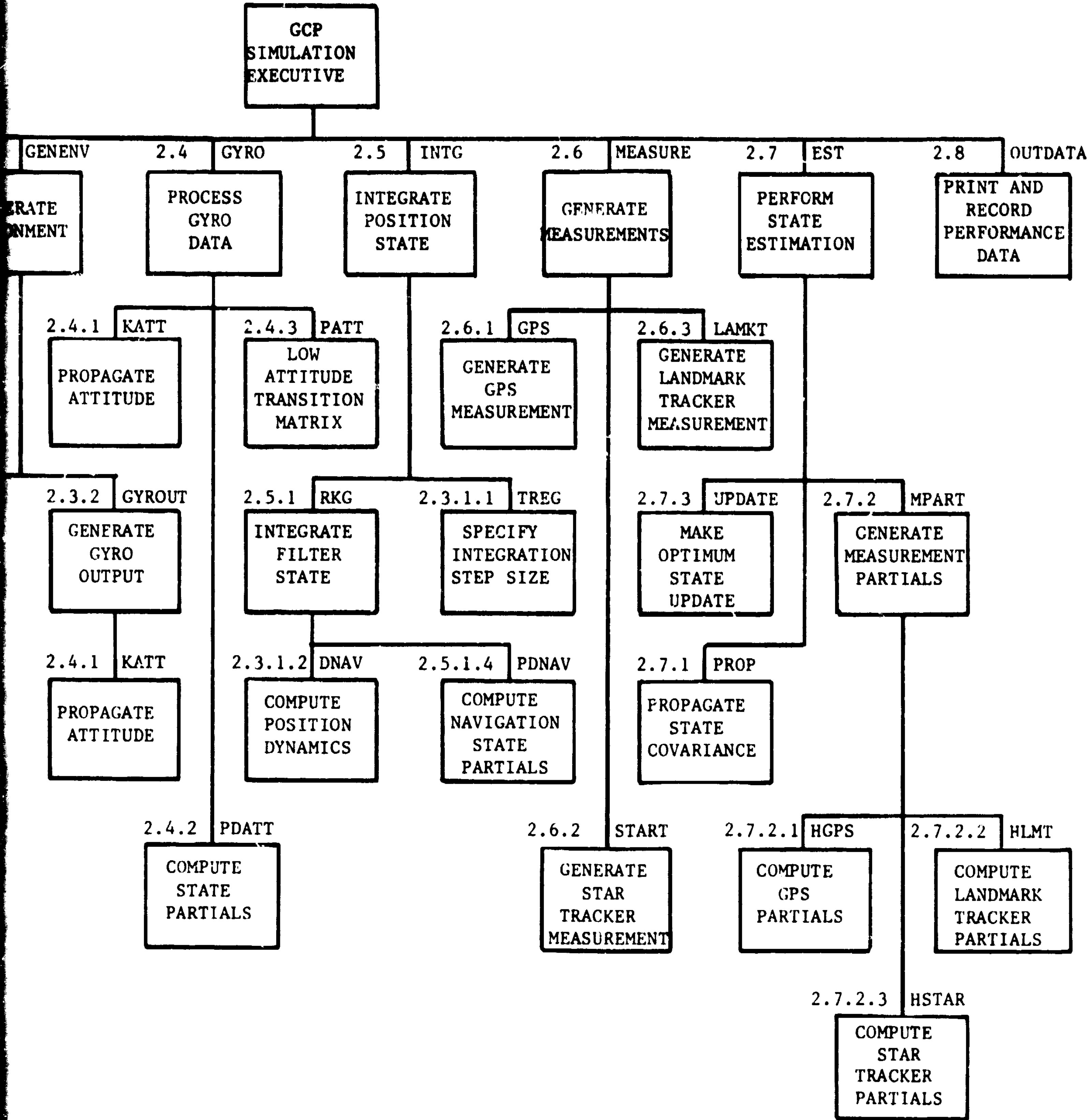


Figure A-2

FOLDOUT FRAME 2

2.0 GCP - Executive Program

The executive program, GCP, is designed to control the entire simulation from data initialization to the end of a run at TSTOP. The main program provides for two modes of operation. The first mode models the measurements from each sensor whereas the second mode replaces the modeled GCP measurements with actual measurements taken from landsat data. Figure A-3 is a VCLR describing the process.

At the top level, the two modes of operation appear to be identical and will be discussed as such. First, simulator initialization is performed within INDATA. This process consists of loading the epoch state and time, and initializing all noise parameters, and setting up the measurement sequence tables. The sequencer, GCPSEQ, determines the type and time of the next measurement to be made. The types of measurements available are GPS, Star Tracker, and landmark tracker. The true vehicle position state is then propagated forward to measurement time by GENENV. GENENV also simulates the gyro output in small increments up to measurement time at TMEAS, and propagates the true attitude state. Within this propagation loop the simulated gyro outputs are processed by GYRO to compensate for gyro bias, scale factor, non-orthogonality, and misalignment, and to update the quaternion. In the second mode of operation this module controls the scrolling of imagery across a video monitor to simulate the sensor front end.

Following true state propagation the appropriate measurement is generated by MEASURE. MEASURE uses a geometry model and the true vehicle state to generate on ideal measurement. The ideal measurement is then corrupted with bias, noise, and misalignment, and compensated for knowledge of these terms. The primary difference between the two modes of operation occurs during measurement generation. In the second mode, the measurement generation module, MEAS, interacts with a separate program running on the PDP 11/70 which performs the GCP extraction.

The estimated state is propagated to measurement time by the module INTG. This propagation is actualy part of the extended Kalman filter, but is kept in a separate module for efficiency. INTG also propagates the state transition matrix and the process noise forward in time.

EST uses the simulated measurement, the propagated state estimate, the process noise, and the state transition matrix to generate a new state estimate based on the latest measurement. The new state estimate and covariance matrix are printed, and the entire process beginning with measurement sequencing is repeated until the end of a simulation run identified by TSTOP. Prior to exiting, a plot file is created so that plots of state error, covariance, and measurement residual, may be obtained.

GCPSIM Executive

- Read Initial Conditions, Sequence Tables, and Control Parameters
- Print Initial Conditions, Sequence Tables, and Control Parameters
- Do Case Simulation Mode =

1) Modeled GCP Measurements

- GCPSEQ
  Sequence Operations, Determine Type & Time of Next Measurement
- Set New Measurement Flag to TRUE
  - GENENV
    Integrate Actual State to Measurement, Generate Gyro Output, Print Gyro Output
  - GYRO
    Process Gyro Data
- Do Until Measurement Time
  TIME.EQ.TMEAS
- MEASURE
  Generate Measurement & Print
  - INTG
    Integrate Filter State to Measurement Time
- Do Until Measurement Time
  TIS.EQ.TMEAS
- EST
  Perform State Estimation
- ICALL = 0 (Measurement Output)
- OUTDATA
  Print Output Data
- Do Until Simulation End
  TIME.EQ.TSTOP
- GCPPLOT
  Generate Tape Plot
- End

2) Reconstructed GCP Measurements

- GCPSEQ
  Sequence Operations, Determine Type & Time of Next Measurement
- Set New Measurement Flag to TRUE
  - ENVIR
    Scroll Image Data to Measurement Time
  - ENVIR
    Integrate Actual State to Measurement Time, Generate Gyro Output
  - GYRO
    Process Gyro Output
- Do Until Measurement Time
  TIME.EQ.TMEAS
- MEAS
  Generate GPS & ST Measurements, Perform GCP Extraction
  - INTG
    Integrate Filter State to Measurement Time
- Do Until Measurement Time
  TIS.EQ.TMEAS
- EST
  Perform State Estimation
- ICALL = 0
- OUTDATA
  Print Output Data
- Do Until Simulation End
  TIME.EQ.TSTOP
- GCPPLOT
  Generate Tape Plot
- End

Figure A-3 GCP VCLR

```
          PROGRAM GCP
          INCLUDE 'CONTRL.COM'
*         COMMON /CONTRL/ MOP,TINT
*         REAL*8 TINT
* C
* C               PROGRAM CONTROL DESCRIPTORS FOR MULTIPLE RUNS
* C
* C                     MOP       MODE OF OPERATION
* C                                1 = PREFLIGHT SIMULATION
* C                                2 = POSTFLIGHT SIMULATION
* C                                3 = MONTE CARLO SIMULATION
* C               TINT       NUMBER OF SECONDS OF FULL OPERATION PER CYCLE
* C
          INCLUDE 'TIME.COM'
        * C
        *       COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
        *     .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
        *       REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
        *     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
        * C
        * C             THESE ARE THE TIME REFERENCE FRAMES
        * C
        * C                  TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
        * C                  TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
        * C                  TSTOP     RUN TERMINATION TIME (SEC)
        * C                  TIA       ATTITUDE INTEGRATION TIME (SEC)
        * C                  DEL          "        "     STEP SIZE (SEC)
        * C                  TIN       POSITION INTEGRATION TIME (SEC)
        * C                  DTN          "        "     STEP SIZE (SEC)
        * C                  DATEO     DATE OF FLIGHT EPOCH (JD)
        * C                  DATER     DATE OF 1950 EPOCH (JD)
        * C                  TZERO     START TIME IN SECS. SINCE DATEO
        * C                  TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
        * C                  TIS       REAL WORLD REFERENCE TIME (SEC)
        * C                  TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
        * C                  DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
        * C                            TOO LARGE AT MEASUREMENT TIME
        * C                  TPRINT    TIME FOR PRINT (SEC)
        * C                  DTPRINT   INCREMENT ON TPRINT (SEC)
        * C
                INCLUDE 'ASTATE.COM'
        * C
        *       COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
        *       REAL*8 DE,E,WD,SF,D,DD
        * C
        * C             ATTITUDE STATE AND CONSIDERED PARAMETERS
        * C
        * C                  D         DIFERENTIAL OF QUATERNIONS
        * C                  E         QUATERNIONS
        * C                  WD        GYRO DRIFT RATE (RAD/SEC)
        * C                  SF        GYRO SCALE FACTOR
        * C                  D         GYRO NON-ORTHOGANALITY (RAD)
        * C                  DD        GYRO RELATIVE ORIENTATION (RAD)
        * C
                INCLUDE 'ROTAT.COM'
        *       COMMON /ROTAT / DTHR(3),DTHEM(3),DTHE(3)
        *       REAL*8 DTHR,DTHEM,DTHE
```

```
        * C
        * C             GYRO ATTITUDE PARAMETERS
        * C
        * C                  DTHR       REAL WORLD GYRO DATA (RAD)
        * C                  DTHEM      FILTER WORLD GYRO DATA (RAD)
        * C                  DTHE       FILTER WORLD COMPENSATED GYRO DATA (RAD)
        * C
0014              INCLUDE 'NOIS_.COM'
0015    *       COMMON /NOISE/ BWD(3),SWD(3),BSF(3),SSF(3),BD(3),SD(3)
        *      .         ,BDD(3),SDD(3),SRM,BRE,SRE
0016    *       REAL*8 BWD,SWD,BSF,SSF,BD,SD,BDD,SDD,SRM,BRE,SRE
        * C
        * C             REAL WORLD  GYRO MEASUREMENT ERRORS
        * C
        * C     BWD  SWD        "       "       GYRO DRIFT (RAD / SEC)
        * C     BSF  SSF        "       "       GYRO SCALE FACTOR
        * C     BD   SD         "       "       GYRO NONORTHOGONALITY (RAD)
        * C     BDD  SDD        "       "       GYRO RELATIVE ORIENTATION (RAD)
        * C
0017              INCLUDE 'NSTATE.COM'
        * C
0018    *       COMMON /NSTATE/ XD(6),X(6),RADM,RADE
0019    *       REAL*8 XD,X,RADM,RADE
        * C
        * C              POSITION STATE AND CONSIDERED PARAMETERS
        * C
        * C                   XD         STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
        * C                   X          STATE POSITION PARAMETERS (KM AND KM/SEC)
        * C                   RADM       RADIUS OF THE MOON (KM)
        * C                   RADE       EARTH DETECTABLE RADIUS (KM)
        * C
        * C
0020              INCLUDE 'DEBUG.COM'
0021    *       COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                  I.NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                  IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
0022              INCLUDE 'PHIA.COM'
0023    *       COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
        *      .        COVA(16,16),POA(16,16),QMAX
0024    *       REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
        * C
        * C             THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
        * C
        * C                  PA         ATTITUDE STATE TRANSITION MATRIX
        * C                  TA             "     PARAMETER TRANSITION MATRIX
        * C                  PDA            "     DERIVATIVE OF TRANSITION MATRICES
        * C                  PHIA           "     AGGREGATE TRANSITION MATRIX
        * C                  COVA           "     NEW COVARIANCE MATRIX
        * C                  POA            "     PREVIOUS COVARIANCE MATRIX
        * C                  QMAX           "     COVARIANCE NORM MAX
        * C
0025              INCLUDE 'PHIN.COM'
0026    *       COMMON /PHIN/ PN(6,6),PDN(6,6),PHIN(6,6),COVN(6,6),
```

```
   *              PON(6,6)
   *        REAL*8 PN,PDN,PHIN,COVN,PON
   * C
   * C             THESE ARE THE NAVIGATION TRANSITION AND COVARIANCE ARRAYS
   * C
   * C                   PN        POSITION STATE TRANSITION MATRIX
   * C                   PDN           "     DERIVATIVE OF TRANSITION MATRIX
   * C                   PHIN          "     AGGREGATE TRANSITION MATRIX
   * C                   COVN          "     NEW COVARIANCE MATRIX
   * C                   PON           "     PREVIOUS COVARIANCE MATRIX
   * C
               INCLUDE 'ARRAYS.COM'
   * C
   *        COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
   *             .T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
   *        REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
   * C
   * C             THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
   * C
   * C                   T1 - T4      SINGLE DIMENSION ARRAYS
   * C                   T11 - T77    DUAL DIMENSIONED ARRAYS
   * C                   T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
   * C
               INCLUDE 'UPDT.COM'
   * C
   *        COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
   *        REAL*8 QN,QA,Q,QDOT
   * C
   * C             STATE ESTIMATION PARAMETERS
   * C
   * C                   QN        NAV. DYN. NOISE COVARIANCE DIAGONAL
   * C                   QA        MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
   * C                   Q         CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
   * C                   QDOT      DIFFERENTIAL OF Q
   * C
               INCLUDE 'RESIDUALS.COM'
   * C*************************************************************************
   * C                                                                        *
   * C--- RESIDUALS.COM CONTAINS THE RESIDUALS VALUES FOR THE GPS,            *
   * C--- LANDMARK TRACKER, AND STAR TRACKER MEASUREMENTS                     *
   * C                                                                        *
   * C*************************************************************************
   * C
   *           COMMON /RESIDUALS/DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,
   *       *               DXMGPS(6)
   *           REAL*8 DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,DXMGPS
   * C
               INCLUDE 'TITL .COM'
   *           COMMON /TITLE/ ATITLE(40)
   *           LOGICAL*1 ATITLE
   * C
   * C               ATITLE IS THE TITLE PRINTED AT EACH PRINT TIME
   * C               AS WELL AS THE TOP OF EACH PLOT
   * C
               LOGICAL FLAG
     C         LOGICAL*1 ATITLE
               DIMENSION IDTHEM(3)
```

```
0042          OPEN(UNIT=6,R CORDSIZE=132,TYPE='NEW')
0043          OPEN(UNIT=12,RECORDSIZE=780,TYPE='NEW')
        C     WRITE(5,2)
        C2    FORMAT(' INPUT RUN TITLE (MAX. 40 CHAR) : ',$)
        C     READ(5,4)(ATITLE(I),I=1,40)
0044    4     FORMAT(40A1)
0045    5     FORMAT(///,5X,40A1)
0046          OPEN(UNIT=13,NAME='GAIN.DAT',RECORDSIZE=98,TYPE='NEW')
        C************************************************************************
        C                INITIALIZE DATA
        C************************************************************************
0047          CALL INDATA
0048          WRITE(12,4)(ATITLE(I),I=1,40)
0049          WRITE(6,5)(ATITLE(I),I=1,40)
0050          ICALL = 0
        C************************************************************************
        C                PRINT INITIAL PARAMETERS
        C************************************************************************
0051          CALL OUTDATA(ICALL)
        C************************************************************************
        C                SET UP FORCE MODELS
        C************************************************************************
0052          CALL DNAV(TIM.,X,XD,IDUMMY)
        C************************************************************************
        C                INITIALIZE OPERATING FLAG TO TRUE
        C************************************************************************
0053          ISKIPN=0
0054          GO TO (10,40),MOP
        C************************************************************************
        C                PREFLIGHT SIMULATION
        C                (SIMULATED GCP MEASUREMENTS)
        C************************************************************************
0055    10    CALL GCPSEQ              ! SEQUENCE MEASUREMENTS
0056          FLAG = .FALSE.            ! INITIALIZE NEW MEAS. FLAG
        C************************************************************************
        C                COMPUTE TIME OF MEASUREMENT
        C
        C
        C                PROPAGATE TRUE VEHICLE STATE AND PROCESS
        C                GYRO DATA TO MEASUREMENT TIME
        C************************************************************************
0057    20    CALL GENENV(FLAG)        ! GENERATE GYRO OUTPUT
0058          CALL GYRO
0059          IF(TIME + .01 .LT. TMEAS) GO TO 20
        C************************************************************************
        C                GENERATE MEASUREMENT
        C************************************************************************
0060          CALL MEASURE
        C************************************************************************
        C                INTEGRATE FILTER STATE TO MEASUREMENT TIME
        C************************************************************************
0061    30    CALL INTG
0062          IF(TIN + .01 .LT. TMEAS) GO TO 30
        C************************************************************************
        C                PERFORM STATE ESTIMATION
        C************************************************************************
0063          CALL EST
```

```
      ICALL = 0
C**********************************************************************
C                 PRINT OUTPUT DATA
C**********************************************************************
      CALL OUTDATA(ICALL)
C**********************************************************************
C                 PROCESS MEASUREMENTS UNTIL TIME=TSTOP
C**********************************************************************
      IF(TIME.LT.TSTOP) GO TO 10
C**********************************************************************
C                 GENERATE PLOT TAPE
C**********************************************************************
      CALL PLTDATA
      GO TO 110
C**********************************************************************
C                 POSTFLIGHT SIMULATION
C**********************************************************************
40    IGCP = .FALSE.
      CALL GCPSEQ
      FLAG = .FALSE.
C**********************************************************************
C                 COMPUTE TIME OF MEASUREMENT
C
C
C                 PROPAGATE TRUE VEHICLE STATE AND PROCESS
C                 GYRO DATA TO MEASUREMENT TIME
C
C**********************************************************************
220   CALL GENENV(FLAG)                 ! GENERATE GYRO OUTPUT
      CALL GYRO
      IF(TIME + .01 .LT. TMEAS) GO TO 220
C**********************************************************************
C                 GENERATE MEASUREMENT
C**********************************************************************
CCALL MEAS
C**********************************************************************
C                 INTEGRATE FILTER STATE TO MEASUREMENT TIME
C**********************************************************************
230   CALL INTG
      IF(TIN + .01 .LT. TMEAS) GO TO 230
C**********************************************************************
C                 PERFORM STATE ESTIMATION
C**********************************************************************
      CALL EST
      ICALL = 0
C**********************************************************************
C                 PRINT OUTPUT DATA
C**********************************************************************
      CALL OUTDATA(ICALL)
C**********************************************************************
C                 PROCESS MEASUREMENTS UNTIL TIME=TSTOP
C**********************************************************************
      IF(TIME .LT. TSTOP) GO TO 40
C**********************************************************************
C                 GENERATE PLOT TAPE
C**********************************************************************
      CALL PLTDATA
```

```
110     CONTINUE
        CLOSE(UNIT=6,DISPOSE='SAVE')
        CLOSE(UNIT=13,DISPOSE='SAVE')
        CLOSE(UNIT=12,DISPOSE='SAVE')
        CALL EXIT
        END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	454	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	24	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	184	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	LONG
3	CONTRL	12	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	ASTATE	160	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	ROTAT	72	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	NOISE	216	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
9	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
10	PHIA	7176	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
11	PHIN	1440	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
12	ARRAYS	1096	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
13	UPDT	752	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
14	RESIDUALS	96	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
15	TITLE	40	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GCP

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
7-000000C8	R*8	BRE	4-00000038	R*8	DATEO	4-00000070	R*8	DATER	4-00000020	R*8	DEL
4-00000068	R*8	DTA	4-00000030	R*8	DTN	4-00000080	R*8	DTPRINT	14-00000000	R*8	DZHLM
14-00000010	R*8	DZHST1	14-00000020	R*8	DZHST2	14-00000008	R*8	DZVLM	14-00000018	R*8	DZVST1
14-00000028	R*8	DZVST2	2-0000000C	L*4	FLAG	2-00000010	I*4	I	2-00000014	I*4	ICALL
9-00000004	I*4	IDEBUG	2-00000018	I*4	IDUMMY	9-00000000	I*4	IENTER	2-00000020	I*4	IGCP
2-0000001C	I*4	ISKIPN	3-00000000	I*4	MOP	10-00001C00	R*8	QMAX	8-00000068	R*8	RADE
8-00000060	R*8	RADM	7-000000D0	R*8	SRE	7-000000C0	R*8	SRM	4-00000018	R*8	TIA
4-00000000	R*8	TIME	4-00000028	R*8	TIN	3-00000004	R*8	TINT	4-00000058	R*8	TIS
4-00000060	R*8	TISN	4-00000048	R*8	TMEAS	4-00000008	R*8	TNEXT	4-00000078	R*8	TPRINT
4-00000050	R*8	TRACK	4-00000010	R*8	TSTOP	4-00000040	R*8	TZERO			

ARRAYS

Address	Type	Name	Bytes	Dimensions
15-00000000	L*1	ATITLE	40	(40)
7-00000060	R*8	BD	24	(3)
7-00000090	R*8	BDD	24	(3)
7-00000030	R*8	BSF	24	(3)
7-00000000	R*8	BWD	24	(3)
10-00000C00	R*8	COVA	2048	(16, 16)
11-00000360	R*8	COVN	288	(6, 6)
5-00000070	R*8	D	24	(3)
5-00000088	R*8	DD	24	(3)
5-00000000	R*8	DE	32	(4)
6-00000030	R*8	DTHE	24	(3)
6-00000018	R*8	DTHEM	24	(3)
6-00000000	R*8	DTHR	24	(3)
14-00000030	R*8	DXMGPS	48	(6)
5-00000020	R*8	E	32	(4)
2-00000000	I*4	IDTHEM	12	(3)
10-00000000	R*8	PA	128	(4, 4)
10-00000200	R*8	PDA	512	(4, 16)
11-00000120	R*8	PDN	288	(6, 6)
10-00000400	R*8	PHIA	2048	(16, 16)
11-00000240	R*8	PHIN	288	(6, 6)
11-00000000	R*8	PN	288	(6, 6)
10-00001400	R*8	POA	2048	(16, 16)
11-00000480	R*8	PON	288	(6, 6)
13-00000080	R*8	Q	288	(6, 6)
13-00000030	R*8	QA	128	(16)
13-00000100	R*8	QDOT	288	(6, 6)
13-00000000	R*8	QN	48	(6)
7-00000078	R*8	SD	24	(3)
7-000000A8	R*8	SDD	24	(3)
5-00000058	R*8	SF	24	(3)
7-00000048	R*8	SSF	24	(3)
7-00000018	R*8	SWD	24	(3)
12-00000000	R*8	T1	24	(3)
12-00000098	R*8	T11	72	(3, 3)
12-00000018	R*8	T2	24	(3)
12-00000030	R*8	T3	24	(3)
12-000000E0	R*8	T33	72	(3, 3)
12-00000048	R*8	T4	80	(10)
12-00000128	R*8	T44	128	(4, 4)
12-000003E8	R*8	T5	32	(4)
12-00000408	R*8	T6	32	(4)
12-000001A8	R*8	T66	288	(6, 6)
12-00000428	R*8	T7	32	(4)
12-000002C8	R*8	T77	288	(6, 6)
10-00000080	R*8	TA	384	(4, 12)
5-00000040	R*8	WD	24	(3)
8-00000030	R*8	X	48	(6)
8-00000000	R*8	XD	48	(6)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000000	4'	1-00000005	5'	0-000000A4	10	0-000000AD	20	0-000000DD	30	0-00000125	40
0-000001A0	110	0-00000131	220	0-0000015A	230						

FUNCTIONS AND SUBROUTINES REFERENCED

DNAV	EST	FOR$CLOSE	FOR$EXIT	FOR$OPEN	GCPSEQ	GENENV	GYRO
INDATA	INTG	MEASURE	OUTDATA	PLTDATA			

Total Space Allocated = 11978 Bytes

COMMAND QUALIFIERS

FORTRAN /L GCP

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time:	3.00 seconds
Elapsed Time:	12.04 seconds
Page Faults:	196
Dynamic Memory:	51 pages

2.1 INITIALIZE DATA MODULE (INDATA)

The Initialize Data module shall perform the basic functions of the telemetry up-link to the spacecraft. It will initialize and reinitialize both attitude and navigation states; the lunar ephemeris, the command sequence tables, and the time reference. When the sequence tables are reinitialized, a flag is set and the sequencing reverts to the beginning of the tables.

A data file is read so that only changes to the standard base need to be input. In addition to inputing data this module will transform attitude data in terms of Euler angles (more convential) to identical attitude data in terms of quaternions.

The attitude parameter transformation is;

$$e_0 = \cos\frac{x}{2}\ \cos\frac{\theta}{2}\ \cos\frac{\phi}{2} + \sin\frac{x}{2}\ \sin\frac{\theta}{2}\ \sin\frac{\phi}{2}$$

$$e_1 = \cos\frac{x}{2}\ \cos\frac{\theta}{2}\ \sin\frac{\phi}{2} - \sin\frac{x}{2}\ \sin\frac{\theta}{2}\ \cos\frac{\phi}{2}$$

$$e_2 = \cos\frac{x}{2}\ \sin\frac{\theta}{2}\ \cos\frac{\phi}{2} + \sin\frac{x}{2}\ \cos\frac{\theta}{2}\ \sin\frac{\phi}{2}$$

$$e_3 = \sin\frac{x}{2}\ \cos\frac{\theta}{2}\ \cos\frac{\phi}{2} - \cos\frac{x}{2}\ \sin\frac{\theta}{2}\ \sin\frac{\phi}{2}$$

The coordinate tranformation for the covariance matrix (P) is computed by;

$$P_{e_0} - e_3 = \Phi P \Phi^T$$

Where, the transition matrix Φ is computed from the above quaternion - Euler angle relationship

$$\Phi = \begin{bmatrix} \frac{\partial e_0}{\partial x} & \frac{\partial e_0}{\partial \theta} & \frac{\partial e_0}{\partial \phi} \\ \frac{\partial e_1}{\partial x} & \frac{\partial e_1}{\partial \theta} & \frac{\partial e_1}{\partial \phi} \\ \frac{\partial e_2}{\partial x} & \frac{\partial e_2}{\partial \theta} & \frac{\partial e_2}{\partial \phi} \\ \frac{\partial e_3}{\partial x} & \frac{\partial e_3}{\partial \theta} & \frac{\partial e_3}{\partial \phi} \end{bmatrix}$$

INDATA VCLR

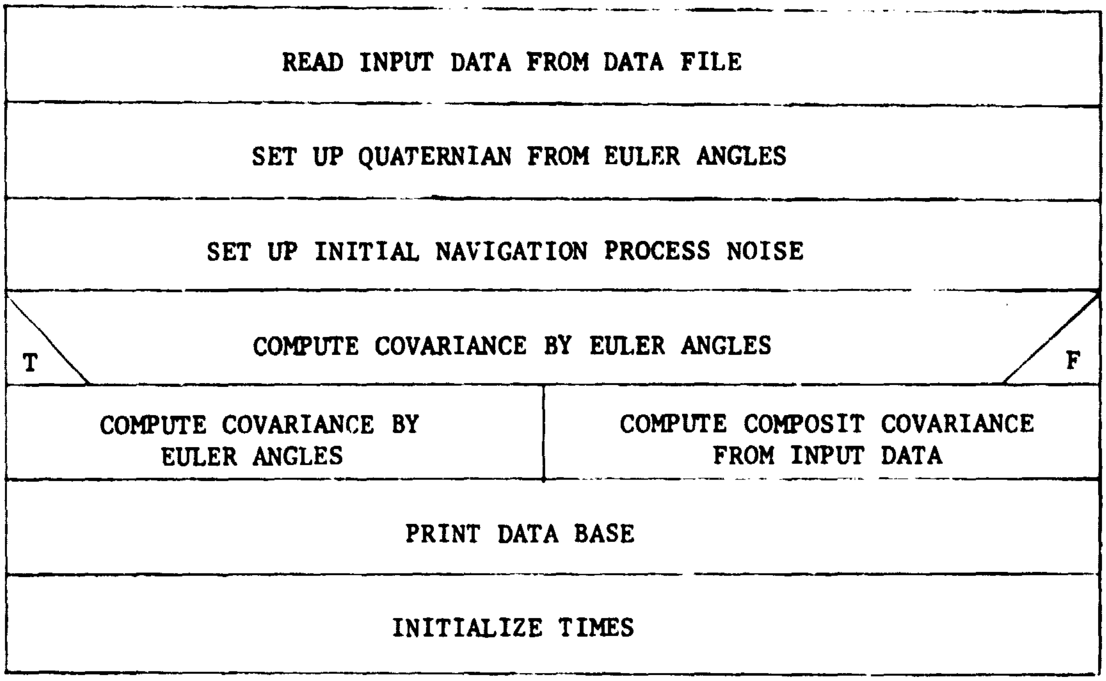


Figure A-4

```
      SUBROUTINE INDATA
      INCLUDE 'DEBUG.COM'
      COMMON /DEBUG/ IENTER,IDEBUG
C
C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
C
C                    I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
C                    IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
C
      INCLUDE 'CONTRL.COM'
      COMMON /CONTRL/ MOP,TINT
      REAL*8 TINT
C
C             PROGRAM CONTROL DESCRIPTORS FOR MULTIPLE RUNS
C
C                    MOP        MODE OF OPERATION
C                               1 = PREFLIGHT SIMULATION
C                               2 = POSTFLIGHT SIMULATION
C                               3 = MONTE CARLO SIMULATION
C             TINT        NUMBER OF SECONDS OF FULL OPERATION PER CYCLE
C
      INCLUDE 'ENVIR.COM'
      COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
      REAL*8 STATE,PROFILE
C
C             REAL WORLD STATE PARAMETERS
C
C                    STATE      STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
C                    PROFILE    ATTITUDE PROFILE-TIME (SEC) VS
C                               INERTIAL ANGULAR RATES (RAD/SEC)
C                    INIT       INTEGRATION INITIALIZATION KEY (-1)
C
      INCLUDE 'GFPART.COM'
      COMMON/GFPART/ FA(4,4),EA(4,12),FN(6,6)
      REAL*8 FA,EA,FN
C
C             MEASUR MENT AND STATE PARTIALS
C
C                    FA         ATTITUDE STATE PARTIALS
C                    EA            "     CONSIDERED PARAMETERS PARTIALS
C                    FN            "     STATE PARTIALS
C
      INCLUDE 'NOISE.COM'
      COMMON /NOISE/ BWD(3),SWD(3),BSF(3),SSF(3),BD(3),SD(3)
     .              ,BDD(3),SDD(3),SRM,BRE,SRE
      REAL*8 BWD,SWD,BSF,SSF,BD,SD,BDD,SDD,SRM,BRE,SRE
C
C             REAL WORLD  GYRO MEASUREMENT ERRORS
C
C     BWD  SWD          "         "      GYRO DRIFT (RAD / SEC)
C     BSF  SSF          "         "      GYRO SCALE FACTOR
C     BD   SD           "         "      GYRO NONORTHOGONALITY (RAD)
C     BDD  SDD          "         "      GYRO RELATIVE ORIENTATION (RAD)
C
      INCLUDE 'TARGETS.COM'
      COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
      LOGICAL JFLAG
```

```
00800   0019  •       REAL*8 PI,TPI
00800         • C
00800         • C             MEASUR MENT SPECIFICATIONS
00800         • C
00800         • C                   MTYPE     MEASUREMENT TYPE
00800         • C                   JFLAG     SET FOR STAR OBSTRUCTION
00800         • C                   MCODE      "    "  MEASUREMENT PROCESSING
00800         • C                   PI        PI
00800         • C                   TPI       2*PI
00800         • C
00800   0020          INCLUDE 'ASTATE.COM'
00900         • C
00900   0021  •       COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
00900   0022  •       REAL*8 DE,E,WD,SF,D,DD
00900         • C
00900         • C             ATTITUDE STATE AND CONSIDERED PARAMETERS
00900         • C
00900         • C                   D         DIFERENTIAL OF QUATERNIONS
00900         • C                   E         QUATERNIONS
00900         • C                   WD        GYRO DRIFT RATE (RAD/SEC)
00900         • C                   SF        GYRO SCALE FACTOR
00900         • C                   D         GYRO NON-ORTHOGANALITY (RAD)
00900         • C                   DD        GYRO RELATIVE ORIENTATION (RAD)
00900         • C
00900   0023          INCLUDE 'FILTER.COM'
01000   0024  •       COMMON/FILTER/ IPN(6),IPA(16)     !IPN WAS IPN(11)   JACK
01000         • C
01000         • C             FILTER DATA CONSTANTS
01000         • C
01000         • C                   IPN       ARRAY INDEX OF ESTIMATED POS PARAMETERS
01000         • C                   IPA         "      "          "     ATT PARAMETERS
01000         • C
01000   0025          INCLUDE 'SEQU.COM'
01100   0026  •       COMMON /SEQU/ I S(25),ITS(4,5),IAA(3,4),IFLAG
01100         •      .         ,DTST,DTGPS,IDELAY(11)
01100   0027  •       REAL*8 DTST,DTGPS
01100   0028  •       LOGICAL IFLAG
01100         • C
01100         • C             SEQUENCE OF EVENTS TABLES
01100         • C
01100         • C                   I S       EVENT SCHEDULAR
01100         • C                   ITS       CYCLE SEQUENCER
01100         • C                   IAA       ALTERNATE ACTIONS
01100         • C                   IFLAG     SET WHEN TARGET IS OBSTRUCTED
01100         • C
01100   0029          INCLUDE 'TMAT.COM'
01200   0030  •       COMMON /TMAT/ A(3,3),B(3,3),C(3,3),EM(4,3)
01200   0031  •       REAL*8 A,B,C,EM
01200         • C
01200         • C             TRANSFORMATION MATRICES
01200         • C
01200         • C                 A         INERTIAL TO BODY AXES
01200         • C                 B         GYRO TO BODY AXES
01200         • C                 C         GYRO NON-ORTHOGONAL TO GYRO AXES
01200         • C                 EM        BODY TO QUATERNIAN AXES
01200         • C
01200   0032          INCLUDE 'NSTATE.COM'
```

```
C
      COMMON /NSTATE/ XD(6),X(6),RADM,RADE
      REAL*8 XD,X,RADM,RADE
C
C             POSITION STATE AND CONSIDERED PARAMETERS
C
C                   XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
C                   X         STATE POSITION PARAMETERS (KM AND KM/SEC)
C                   RADM      RADIUS OF THE MOON (KM)
C                   RADE      EARTH DETECTABLE RADIUS (KM)
C
C
      INCLUDE 'ARRAYS.COM'
C
      COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
     *               ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
      REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
C
C            THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
C
C                 T1 - T4      SINGLE DIMENSION ARRAYS
C                 T11 - T77    DUAL DIMENSIONED ARRAYS
C                 T11          DUAL ARRAY; OFF DIAGONAL SET TO ZERO
C
      INCLUDE 'TIME.COM'
C
      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
     *             ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
     *       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
C
C            THESE ARE THE TIME REFERENCE FRAMES
C
C                 TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
C                 TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
C                 TSTOP      RUN TERMINATION TIME (SEC)
C                 TIA        ATTITUDE INTEGRATION TIME (SEC)
C                 DEL           "         "         STEP SIZE (SEC)
C                 TIN        POSITION INTEGRATION TIME (SEC)
C                 DTN           "         "         STEP SIZE (SEC)
C                 DATEO      DATE OF FLIGHT EPOCH (JD)
C                 DATER      DATE OF 1950 EPOCH (JD)
C                 TZERO      START TIME IN SECS. SINCE DATEO
C                 TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
C                 TIS        REAL WORLD REFERENCE TIME (SEC)
C                 TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
C                 DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
C                            TOO LARGE AT MEASUREMENT TIME
C                 TPRINT     TIME FOR PRINT (SEC)
C                 DTPRINT    INCREMENT ON TPRINT (SEC)
C
      INCLUDE 'CONST.COM'
      COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
      REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
C
C            PROGRAM CONSTANTS
C
```

```
01600          * C                 ATM       S/C AREA TO MASS RATIO (METERS/KG)
01600          * C                 RBM       OBSTRUCTION RADIUS OF THE MOON (KM)
01600          * C                 RBE            "       "           EARTH (KM)
01600          * C                 RBO            "       "           SUN (KM)
01600          * C                 R 2       SQUARE OF THE EARTHS RADIUS (KM 2)
01600          * C                 RM2         "          LUNAR RADIUS (KM 2)
01600          * C                 UM        LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
01600          * C                 US        SOLAR          "         "        "
01600          * C                 U         EARTH          "         "        "
01600          * C                 J2,J3,J4  ZONAL GRAVITATIONAL HARMONIC TERMS
01600          * C                 DTU       REGULARIZED TIME STEP SIZE (SEC)
01600          * C                 PK1       SOLAR PRESSURE CONSTANT
01600          * C
01600   0044          INCLUDE 'UPDT.COM'
01700          * C
01700   0045   *      COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
01700   0046   *      REAL*8 QN,QA,Q,QDOT
01700          * C
01700          * C         STATE ESTIMATION PARAMETERS
01700          * C
01700          * C                 QN        NAV. DYN. NOISE COVARIANCE DIAGONAL
01700          * C                 QA        MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
01700          * C                 Q         CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
01700          * C                 QDOT      DIFFERENTIAL OF Q
01700          * C
01700   0047          INCLUDE 'PLOT.COM'
01800          * C
01800   0048   *      COMMON /PLOT/ TP1,TP2
01800   0049   *      REAL*8 TP1,TP2
01800          * C
01800          * C         PLOTING INFORMATION
01800          * C
01800          * C                 TP1       LOWER ABSCISSA VALUE - TIME (MIN)
01800          * C                 TP2       UPPER    "        "       "
01800          * C
01800   0050          INCLUDE 'PHIA.COM'
00100   0051   *      COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
00200          *     .        COVA(16,16),POA(16,16),QMAX
00300   0052   *      REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
00400          * C
00500          * C         THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
00600          * C
00700          * C                 PA        ATTITUDE STATE TRANSITION MATRIX
00800          * C                 TA           "     PARAMETER TRANSITION MATRIX
00900          * C                 PDA          "     DERIVATIVE OF TRANSITION MATRICES
01000          * C                 PHIA         "     AGGREGATE TRANSITION MATRIX
01100          * C                 COVA         "     NEW COVARIANCE MATRIX
01200          * C                 POA          "     PREVIOUS COVARIANCE MATRIX
01300          * C                 QMAX         "     COVARIANCE NORM MAX
01400          * C
01900   0053          INCLUDE 'PHIN.COM'
02000   0054   *      COMMON /PHIN/ PN(6,6),PDN(6,6),PHIN(6,6),COVN(6,6),
02000          *     .        PON(6,6)
02000   0055   *      REAL*8 PN,PDN,PHIN,COVN,PON
02000          * C
02000          * C         THESE ARE THE NAVIGATION TRANSITION AND COVARIANCE ARRAYS
02000          * C
```

```
02000          * C              PN        POSITION STATE TRANSITION MATRIX
02000          * C              PDN          "      DERIVATIVE OF TRANSITION MATRIX
02000          * C              PHIN         "      AGGREGATE TRANSITION MATRIX
02000          * C              COVN         "      NEW COVARIANCE MATRIX
02000          * C              PON          "      PREVIOUS COVARIANCE MATRIX
02000          * C
02000    0056          INCLUDE 'GCPDAT.COM'
02100    0057    *       COMMON /GCPDAT/ IGCPDT(4,10),LINNUM,TGCP,IDLMAX,GCPN,DTPS
02100    0058    *       REAL*8 TGCP,GCPN,DTPS
02100    0059          INCLUDE 'STARPAR.COM'
02200    0060    *       COMMON /STARPAR/ BS(2,2),SS(2,2),TNS(3,3,2),TBNS(3,3,2),
02200            *     .         BSK(2,2),SSK(2,2),TNSK(3,3,2)
02200    0061    *       REAL*8 BS,SS,TNS,TBNS,BSK,SSK,TNSK
02200          * C
02200          * C      STAR TRACKER PARAMETERS
02200          * C              IN EACH CASE THE LAST SUBSCRIPT REFERS TO THE
02200          * C                      TRACKER USED
02200          * C              BS       = BIAS - ACTUAL (RAD)
02200          * C              SS       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
02200          * C              TNS      = MISALIGNMENT ARRAY - TRANSFORMATION FROM
02200          * C                                 STAR TRACKER TO NOMINAL
02200          * C              TBNS     = ORIENTATION ARRAY - TRANSFORMATION FROM
02200          * C                                 NOMINAL TO BODY
02200          * C              BSK      = BIAS - KNOWLEDGE (RAD)
02200          * C              SSK      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
02200          * C              TNSK     = MISALIGNMENT KNOWLEDGE ARRAY - TRANSFORMATION
02200          * C                                 FROM STAR TRACKER TO NOMINAL
02200          * C
02200    0062          INCLUDE 'COMPOSIT.COM'
02300    0063    *       COMMON /COMPOSIT/ PHI(22,22),QT(22,22),COV(22,22),PO(22,22),
02300            *               IP(22),XT(22),P(22,22)
02300    0064    *      REAL*8 PHI,QT,COV,PO,XT,P
02300          * C
02300          * C              PHI      = COMPOSIT STATE TRANSITION MATRIX
02300          * C              QT       =     "      PROCESS NOISE ARRAY
02300          * C              COV      = NEWEST COVARIANCE ARRAY
02300          * C              PO       = OLD COVARIANCE ARRAY
02300          * C              IP       = ARRAY OF FLAGS INDICATING ESTIMATED AND
02300          * C                                 CONSIDERED PARAMETERS
02300          * C              XT       = COMPOSIT ESTIMATED PLUS CONSIDERED
02300          * C                                 STATE VECTOR
02300          * C              F        = INITIALIZED TRANSITION MATRIX FOR NEXT
02300          * C                                 INTREVAL
02300          * C
02300    0065          INCLUDE 'LMTPAR.COM'
00100    0066    *       COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
00200            *     .         BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
00300    0067    *       REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
00400            *     .         THET
00500          * C
00600          * C
00700          * C       LANDMARK TRACKER PARAMETERS
00800          * C              AL       = ALTITUDE OF LANDMARK (KM)
00900          * C              LON      = LONGITUDE OF LANDMARK (DEG)
01000          * C              LAT      = LATITUDE OF LANDMARK (DEG)
01100          * C              TBNL     = ORIENTATION ARRAY FOR LANDMARK TRACKER
01200          * C                                 NOMINAL TO BODY
```

```
            • C             TNL      = MISALIGNMENT ARRAY - ACTUAL
            • C                               TRACKER TO NOMINAL
            • C             BL       = BIAS - ACTUAL (RAD)
            • C             SL       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
            • C             BKL      = BIAS - KNOWLEDGE (RAD)
            • C             THET     = LOOK ANGLE (RAD)
            • C             SKL      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
            • C             TIEO     = INITIAL EARTH FIXED TO INERTIAL
            • C                               TRANSFORMATION
            • C             TNLK     = MISALIGNMENT ARRAY KNOWLEDGE
            • C                               TRACKER TO NOMINAL
            • C             SIGGCP   = POSITION UNCERTAINTY DUE TO CLOUDS
            • C
      0068           INCLUDE 'MEASOUT.COM'
      0069  •          COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
            •       .           RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
            •       .           EDHS(2),EDVS(2),EDHL,EDVL
      0070  •          REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
            •       .           EDHS,EDVS,EDHL,EDVL
            • C
            • C
            • C        MEASUREMENT OUTPUT PARAMETERS
            • C             MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
            • C                               (KM,KM/SEC)
            • C             EMXG     = ESTIMATED POSITION/VELOCITY STATE
            • C                               MEASUREMENT - HGPS
            • C             RGPS     = STATE MEASUREMENT NOISE COVARIANCE
            • C                               (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
            • C             DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
            • C                               FROM BORESIGHT - START (RAD)
            • C             DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
            • C                               FROM BORESIGHT - START (RAD)
            • C             EDHS     = ESTIMATED STAR MEASUREMENT HORIZONTAL
            • C                               DEVIATION FROM BORESIGHT (RAD)
            • C             EDVS     = ESTIMATED STAR MEASUREMENT VERTICAL
            • C                               DEVIATION FROM BORESIGHT (RAD)
            • C             MS       = STAR MEASUREMENT UNIT VECTOR (SECOND
            • C                               SUBSCRIPT REFERS TO TRACKER) - START
            • C             RS       = STAR MEASUREMENT NOISE COVARIANCE
            • C                               (KNOWLEDGE) - START (RAD**2)
            • C             DHCL     = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
            • C                               FROM BORESIGHT - LAMKT (RAD)
            • C             DVCL     = LANDMARK MEASUREMENT VERTICAL DEVIATION
            • C                               FROM BORESIGHT - LAMKT (RAD)
            • C             EDHL     = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
            • C                               DEVIATION FROM BORESIGHT (RAD)
            • C             EDVL     = ESTIMATED LANDMARK MEASUREMENT VERTICAL
            • C                               DEVIATION FROM BORESIGHT (RAD)
            • C             LMU      = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
            • C             RL       = LANDMARK MEASUREMENT NOISE COVARIANCE
            • C                               (KNOWLEDGE) - LAMKT (RAD**2)
            • C
      0071           INCLUDE 'GPSPAR.COM'
      0072  •          COMMON /GPSPAR/ PB(3),VB(3),PS(3),VS(3),PBK(3),VBK(3),PSK(3),
            •       .           VSK(3)
      0073  •          REAL*8 PB,VB,PS,VS,PBK,VBK,PSK,VSK
            • C
```

```
02600          • C        GPS PARAMETERS
02600          • C                PB       = POSITION BIAS - ACTUAL
02600          • C                VB       = VELOSITY BIAS - ACTUAL
02600          • C                PS       = POSITION NOISE STANDARD DEVIATION - ACTUAL
02600          • C                VS       = VELOCITY NOISE STANDARD DEVIATION - ACTUAL
02600          • C                PBK      = POSITION BIAS - KNOWLEDGE
02600          • C                VBK      = VELOCITY BIAS - KNOWLEDGE
02600          • C                PSK      = POSITION NOISE STANDARD DEVIATION -
02600          • C                                   KNOWLEDGE
02600          • C                VSK      = VELOSITY NOISE STANDARD DEVIATION -
02600          • C                                   KNOWLEDGE
02600          • C
02600    0074           INCLUDE 'PART.COM'
02700    0075  •          COMMON /PART/ PX(22),PY(22),PZ(22),PXD(22),PYD(22),PZD(22),
02700          •                   PDHS(22,2),PDVS(22,2),PDHL(22),PDVL(22)
02700    0076  •          REAL*8 PX,PY,PZ,PXD,PYD,PZD,PDHS,PDVS,PDHL,PDVL
02700          • C
02700          • C          PARTIALS OF THE RESPECTIVE MEASUREMENTS MADE
02700          • C
02700          • C          FOR GPS
02700          • C                PX       = PARTIALS OF X POSITION MEASUREMENT
02700          • C                PY       =    "      "  Y    "          "
02700          • C                PZ       =    "      "  Z    "          "
02700          • C                PXD      =    "      "  X VELOCITY      "
02700          • C                PYD      =    "      "  Y    "          "
02700          • C                PZD      =    "      "  Z    "          "
02700          • C          FOR STAR TRACKER K (K IS THE SECOND PARAMETER)
02700          • C                PDHS     = PARTIALS OF HORIZONTAL DIFLECTION
02700          • C                PDVS     =    "      "  VERTICAL         "
02700          • C          FOR LANDMARK TRACKER
02700          • C                PDHL     = PARTIALS OF HORIZONTAL DIFLECTION
02700          • C                PDVL     =    "      "  VERTICAL         "
02700          • C
02700    0077           INCLUDE 'CLOUD.COM'
00100    0078  •        COMMON /CLOUD/ CLOTBL(12)
00200    0079  •        REAL*8 CLOTBL
00300          • C                  PCNT        THE PERCENTAGE OF CLOUD COVER
00400          • C
02800    0080             INCLUDE 'TITL..COM'
00100    0081  •          COMMON /TITLE/ ATITLE(40)
00200    0082  •          LOGICAL*1 ATITLE
00300          • C
00400          • C                ATITL. IS THE TITLE PRINTED AT EACH PRINT TIME
00500          • C                AS WELL AS THE TOP OF EACH PLOT
00600          • C
02900    0083             INCLUDE 'MODE.COM'
00100    0084  •          COMMON /MODE/ MODE(10)
00200          • C
00300          • C                MODE(1) = LANDMARK TRACKER SWEEP MODE
00400          • C                                  0 = RANDOM
00500          • C                                  1 = FIXED AT INPUT THET
00600          • C                                  2 = NO DEFAULT TO STAR TRACKER
00700          • C                MODE(2) = CLOUD SELECTION MODE
00800          • C                                  0 = RANDOM CLOUD DENSITIES BASED
00900          • C                                          ON INPUT TABLES CLOTBL
01000          • C                                  1 = FIXED DENSITY AT NO CLOUDS
01100          • C                                  2 = NO CLOUDS WITH 100% CLOUD
```

```
C
C                                         COVER FOR A SPECIFIED
C                                         PERIOD (CLOTBL(11,12))
C               MODE(3-10) NOT SPECIFIED AT PRESENT
      DIMENSION TP(4,3),TPT(3,4),PTP(3,3),PQ(4,4),TMP(3,4)
      DIMENSION SDA(16),SDN(6)
C     EQUIVALENCE (T1(1),PSI),(T1(2),THE),(T1(3),PHI)
C
      REAL*8 A1,A2,A3,A4,A5,A6,A7,A8,C1,C2,C3S1,S2,S3,TSTART,PQ,PTP,
     .       SDA,TMP,TP,TPT,C3,S1
C     TAPE5      ATTITUDE PLOT TAPE
C     TAPE7      NAVIGATION PLOT TAPE
C
C
      OPEN(UNIT=4,TYP =‘OLD’)
        READ(4,4) (ATITLE(I),I=1,40)
4       FORMAT(20X,40A1)
      READ(4,60)((A(I,J),I=1,3),J=1,3)
      READ(4,10)AL
      READ(4,10)ATM
      READ(4,60)((B(I,J),I=1,3),J=1,3)
      READ(4,60)(BD(I),I=1,3)
      READ(4,60)(BDD(I),I=1,3)
      READ(4,45)(BKL(I),I=1,2)
      READ(4,45)(BL(I),I=1,2)
      READ(4,10)BRE
      READ(4,45)((BS(I,J),I=1,2),J=1,2)
      READ(4,45)((BSK(I,J),I=1,2),J=1,2)
      READ(4,60)(BSF(I),I=1,3)
      READ(4,60)(BWD(I),I=1,3)
      READ(4,60)((C(I,J),I=1,3),J=1,3)
      READ(4,90)(CLOTBL(I),I=1,10)
        READ(4,45)CLOTBL(11),CLOTBL(12)
      READ(4,115)((COV(I,J),I=1,22),J=1,22)
      READ(4,130)(COVA(I,I),I=1,16)
      READ(4,130)(COVN(I,I),I=1,6)
      READ(4,60)(D(I),I=1,3)
      READ(4,120)DATEO
      READ(4,120)DATER
      READ(4,60)(DD(I),I=1,3)
      READ(4,70)(DE(I),I=1,4)
      READ(4,10)DEL
      READ(4,10)DHCL
      READ(4,45)(DHCS(I),I=1,2)
      READ(4,10)DTA
      READ(4,20)DTGPS,DTN
      READ(4,10)DTPRINT
      READ(4,20)DTPS,DTST
      READ(4,10) DTU
      READ(4,10)DVCL
      READ(4,45)(DVCS(I),I=1,2)
      READ(4,70)(E(I),I=1,4)
      READ(4,70)((EA(I,J),I=1,4),J=1,12)
      READ(4,70)((FA(I,J),I=1,4),J=1,4)
      READ(4,80)((FN(I,J),I=1,6),J=1,6)
      READ(4,140)ICOVDI
      READ(4,200)IDEBUG,IENTER
```

```
      READ(4,166)(IDELAY(I),I=1,11)
      READ(4,140) IDLMAX
      READ(4,180)(IES(I),I=1,25)
      READ(4,210)IFLAG
      READ(4,140)IGCP
      READ(4,160)((IGCPDT(I,J),I=1,4),J=1,10)
      READ(4,140)INIT
      READ(4,195)(IP(I),I=1,22)
      READ(4,190)(IPA(I),I=1,16)
      READ(4,170)(IPN(I),I=1,6)
      READ(4,140)IS
      READ(4,160)((ITS(I,J),I=1,4),J=1,5)
      READ(4,120)J2
      READ(4,120)J3
      READ(4,120)J4
      READ(4,210)JFLAG
      READ(4,10)LAT
      READ(4,60)(LMU(I),I=1,3)
      READ(4,10)LON
      READ(4,140)MCOD
        READ(4,165)(MODE(I),I=1,10)
      READ(4,140)MOP
      READ(4,60)((MS(I,J),I=1,3),J=1,2)
      READ(4,140)MTYP
      READ(4,80)(MX(I),I=1,6)
      READ(4,115)((P(I,J),I=1,22),J=1,22)
      READ(4,70)((PA(I,J),I=1,4),J=1,4)
      READ(4,60)(PB(I),I=1,3)
      READ(4,60)(PBK(I),I=1,3)
      READ(4,70)((PDA(I,J),I=1,4),J=1,16)
      READ(4,115)(PDHL(I),I=1,22)
      READ(4,115)((PDHS(I,J),I=1,22),J=1,2)
      READ(4,80)((PDN(I,J),I=1,6),J=1,6)
      READ(4,115)(PDVL(I),I=1,22)
      READ(4,115)((PDVS(I,J),I=1,22),J=1,2)
      READ(4,115)((PHI(I,J),I=1,22),J=1,22)
      READ(4,110)((PHIA(I,J),I=1,16),J=1,16)
      READ(4,80)((PHIN(I,J),I=1,6),J=1,6)
      READ(4,10)PI
      READ(4,120)PK1
      READ(4,80)((PN(I,J),I=1,6),J=1,6)
      READ(4,115)((PO(I,J),I=1,22),J=1,22)
      READ(4,110)((POA(I,J),I=1,16),J=1,16)
      READ(4,80)((PON(I,J),I=1,6),J=1,6)
      READ(4,90)((PROFILE(I,J),I=1,10),J=1,4)
      READ(4,60)(PS(I),I=1,3)
      READ(4,60)(PSK(I),I=1,3)
      READ(4,115)(PX(I),I=1,22)
      READ(4,115)(PY(I),I=1,22)
      READ(4,115)(PZ(I),I=1,22)
      READ(4,115)(PXD(I),I=1,22)
      READ(4,115)(PYD(I),I=1,22)
      READ(4,115)(PZD(I),I=1,22)
      READ(4,80)((Q(I,J),I=1,6),J=1,6)
      READ(4,110)(QA(I),I=1,16)
      READ(4,80)((QDOT(I,J),I=1,6),J=1,6)
      READ(4,10)QMAX
```

```
      READ(4,80)(QN(I),I=1,6)
      READ(4,115)((QT(I,J),I=1,22),J=1,22)
      READ(4,40)RADE,RADM,RBE,RBM
      READ(4,10)RBO
      READ(4,120)RE2
      READ(4,80)((RGPS(I,J),I=1,6),J=1,6)
      READ(4,70)((RL(I,J),I=1,2),J=1,2)
      READ(4,120)RM2
      READ(4,45)(((RS(I,J,K),I=1,2),J=1,2),K=1,2)
      READ(4,60)(SD(I),I=1,3)
      READ(4,60)(SDD(I),I=1,3)
      READ(4,60)(SF(I),I=1,3)
      READ(4,10)SIGGCP
      READ(4,45)(SKL(I),I=1,2)
      READ(4,45)(SL(I),I=1,2)
      READ(4,10)SRE
      READ(4,60)(SSF(I),I=1,3)
      READ(4,70)((SS(I,J),I=1,2),J=1,2)
      READ(4,70)((SSK(I,J),I=1,2),J=1,2)
      READ(4,80)(STAT (I),I=1,10)
      READ(4,60)(SWD(I),I=1,3)
      READ(4,60)((TBNL(I,J),I=1,3),J=1,3)
      READ(4,60)(((TBNS(I,J,K),I=1,3),J=1,3),K=1,2)
        READ(4,10)THET
      READ(4,60)((TIEO(I,J),I=1,3),J=1,3)
      READ(4,60)((TNL(I,J),I=1,3),J=1,3)
      READ(4,60)((TNLK(I,J),I=1,3),J=1,3)
      READ(4,60)(((TNS(I,J,K),I=1,3),J=1,3),K=1,2)
      READ(4,60)(((TNSK(I,J,K),I=1,3),J=1,3),K=1,2)
      READ(4,60)(T1(I),I=1,3)
      READ(4,60)((T11(I,J),I=1,3),J=1,3)
      READ(4,60)(T2(I),I=1,3)
      READ(4,60)(T3(I),I=1,3)
      READ(4,60)((T33(I,J),I=1,3),J=1,3)
      READ(4,90)(T4(I),I=1,10)
      READ(4,70)((T44(I,J),I=1,4),J=1,4)
      READ(4,70)(T5(I),I=1,4)
      READ(4,70)(T6(I),I=1,4)
      READ(4,80)((T66(I,J),I=1,6),J=1,6)
      READ(4,70)(T7(I),I=1,4)
      READ(4,80)((T77(I,J),I=1,6),J=1,6)
      READ(4,70)((TA(I,J),I=1,4),J=1,12)
      READ(4,30)TIA,TIME,TIN
      READ(4,40)TINT,TIS,TISN,TMEAS
      READ(4,10)TNEXT
      READ(4,30)TP1,TP2,TPI
      READ(4,10)TPRINT
      READ(4,10)TSTOP
      READ(4,10)TZERO
      READ(4,120)UE
      READ(4,120)UM
      READ(4,120)US
      READ(4,60)(VB(I),I=1,3)
      READ(4,60)(VBK(I),I=1,3)
      READ(4,60)(VS(I),I=1,3)
      READ(4,60)(VSK(I),I=1,3)
      READ(4,60)(WD(I),I=1,3)
```

```
      READ(4,115)(XT(I),I=1,22)
      READ(4,80)(X(I),I=1,6)
      READ(4,80)(XD(I),I=1,6)
C
10    FORMAT(20X,F20.10)
20    FORMAT(20X,F20.10,/,20X,F20.10)
30    FORMAT(20X,F20.10,/,20X,F20.10,/,20X,F20.10)
40    FORMAT(20X,F20.10,/,20X,F20.10,/,20X,F20.10,/,20X,F20.10)
45    FORMAT(20X,2F20.10)
60    FORMAT(20X,3F20.10)
70    FORMAT(20X,4F20.10)
80    FORMAT(20X,6F20.10)
90    FORMAT(20X,10F20.10)
100   FORMAT(20X,11F20.10)
110   FORMAT(20X,16F20.10)
115   FORMAT(20X,22F20.10)
120   FORMAT(20X,E20.10)
130   FORMAT(20X,5E20.10)
140   FORMAT(20X,I10)
150   FORMAT(20X,3I10)
160   FORMAT(20X,4I10)
165   FORMAT(20X,10I10)
166   FORMAT(20X,11I10)
170   FORMAT(20X,11I10)
180   FORMAT(20X,12I10)
190   FORMAT(20X,16I10)
195   FORMAT(20X,22I10)
200   FORMAT(20X,I10,/,20X,I10)
210   FORMAT(20X,L7)
      REWIND 7
C***************************************************************************
C     PSI = 361.
C***************************************************************************
      T1(3)=361.                                  ! TEMPORARY FIX; JACK
      DO 400 I=1,16                               ! TEMPORARY FIX
400   SDA(I)=0.                                   ! TEMPORARY FIX
      DO 450 I=1,6                                ! TEMPORARY FIX
450   SDN(I)=0.                                   ! TEMPORARY FIX
      DO 500 I = 1,6
          DO 500 J = 1,6
500           Q(I,J) = 0.
C***************************************************************************
C     IF(PSI.EQ.361.) GO TO 1000
C***************************************************************************
      IF(T1(3) .EQ. 361.) GO TO 1000              ! TEMPORARY FIX; JACK
      C1 = COS(T1(1))
      S1 = SIN(T1(1))
      C2 = COS(T1(2))
      S2 = SIN(T1(2))
      C3 = COS(T1(3))
      S3 = SIN(T1(3))
      A1 = S1*C2*C3
      A2 = C1*S2*S3
      A3 = C1*S2*C3
      A4 = S1*C2*S3
      A5 = C1*C2*S3
      A6 = S1*S2*C3
```

```
      A7 = C1*C2*C3
      A8 = S1*S2*S3
      E(1) = A7+A8
      E(2) = A5-A6
      E(3) = A3+A4
      E(4) = A1-A2
C
 1000 CONTINUE
      DO 1100 I=1,6
 1100 Q(I,I)=QN(I)
      IF(ICOVDI.EQ.0) GO TO 8000
      DO 2200 I=1,6
 2200 COV(I,I)=COVN(I,I)
      DO 3300 I= 1,16
 3300 COV(I+6,I+6)=COVA(I,I)
      IF(SDN(1).EQ.0.) GO TO 2500
      DO 2000  I=1,6
 2000 COVN(I,I) = SDN(I)**2
      DO 2100 I=1,6
 2100 COV(I,I)=COVN(I,I)
 2500 IF(SDA(2).EQ.0.) GO TO 8000
      DO 3000  I=1,16
 3000 COVA(I,I) = SDA(I)**2
      DO 3100 I=1,16
 3100 COV(I+6,I+6)=COVA(I,I)
      DO 4000  I=1,3
      J = I+1
 4000 PTP(I,I) = SDA(J)**2
      TP(1,1) = A2-A1
      TP(2,1) =-A3-A4
      TP(3,1) = A5-A6
      TP(4,1) = A7+A8
      TP(1,2) = A4-A3
      TP(2,2) =-A1-A2
      TP(3,2) = A7-A8
      TP(4,2) =-A5-A6
      TP(1,3) = A6-A5
      TP(2,3) = A7+A8
      TP(3,3) = A1-A2
      TP(4,3) =-A3-A4
      DO 5000  I=1,3
      DO 5000  J=1,4
      TP(J,I) = .5*TP(J,I)
 5000 TPT(I,J) = TP(J,I)
      CALL MATAB(PTP,TP ,TMP,3,3,4)
      CALL MATAB(TP,TMP,PQ,4,3,4)
      DO 6000  I=1,4
      DO 6000  J=1,4
 6000 COVA(I,J) = PQ(I,J)
      DO 7000 I=1,16
      DO 7000 J=1,16
 7000 POA(I,J)=COVA(I,J)
      DO 7500 I=1,6
      DO 7500 J=1,6
 7500 PON(I,J)=COVN(I,J)
 8000 WRITE(6,9000) TSTART,TSTOP,E,X
      WRITE(6,10000) (COVA(I,I),I=1,16)
```

```
0351              WRITE(6,11000) (COVN(I,I),I=1,6)
0352        9000  FORMAT(1H1,5X,46HGROUND CONTROL POINT SIMULATION PROGRAM GCPSIM
                 .//5X,25HRUN START AND STOP TIMES ,2F10.0/5X,
                 .25HINIT QUATERNION VALUES    ,4F10.7/5X,
                 .25HINITIAL POSITION VALUES  ,3E27.14/30X,3E27.14/30X,5E12.4///)
0353        10000 FORMAT(5X,25HATT PARAMETER VARIANCES  ,8E12.4/30X,8E12.4)
0354        11000 FORMAT(5X,25HNAV PARAMETER VARIANCES  ,6E12.4/30X,5E12.4///)
0355              WRITE(6,11200) MOP
0356        11200 FORMAT(/,4X,' MODE OF OPERATION = ',I1)
0357              WRITE(6,11300) UE,US,UM
0358        11300 FORMAT(4X,' GRAVITATIONAL CONSTANTS -- EARTH, SUN, MOON ',4X,
                 . 3E22.14)
0359              WRITE(6,11400) J2, J3, J4
0360        11400 FORMAT(4X,' ZONAL GRAVITATIONAL HARMONIC TERMS -- ',
                 . 'J2, J3, J4',3 22.14)
0361              WRITE(6,11500) RBE, RBO, RBM
0362        11500 FORMAT(4X,' REAL WORLD RADII -- EARTH, SUN, MOON ',11X,3E22.14)
0363              WRITE(6,11600) RADE, RADM
0364        11600 FORMAT(4X,' ESTIMATED RADII  -- EARTH,      MOON ',11X,
                 . E22.14,22X,E22.14,/)
0365              TIME=TSTART
0366              TIN=TSTART
0367              TIA=TSTART
0368              TIS=TSTART
            C*****************************************************************
            C        FOR EOR2, TMEAS READ FROM NAMELIST.  IN ALL OTHER CASES WISH IT
            C        TO EQUAL TSTART.  FOR EOR2, HOWEVER, TMEAS IS A CONSTANT OF 30 SECS.
            C        AND MUST BE READ IN THROUGH NAMELIST.  SEE SPECIAL LOGIC IN EXEC.
            C*****************************************************************
0369              IF(MOP.NE.2) TMEAS = TSTART
0370              WRITE(6,12000) IES
0371              WRITE(6,13000) ITS
0372              WRITE(6,14000) IST,IAA
0373              WRITE(6,15000) IPA,IPN
0374              WRITE(6,16000)
0375              RETURN
0376        12000 FORMAT(20X,16HEVENT SEQUENCING/15X,75H  1  2  3  4  5  6  7  8  9
                 .10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 /15X,25I3///)
0377        13000 FORMAT(20X,13HEVENT CYCLING/25X,50HEVENT START - STOP, MAX NO. CYC
                 .LES, MAX TIME - HRS  /15X,5(4I3,4X)///)
0378        14000 FORMAT(20X,12HACTION TABLE/20X,52HCODE     1  2  3  4  5  6  7  8
                 .9 10 11 12 13 14 15  /20X,18HSTAR SPECIFICATION/3(29X,11I3/)20X,
                 .16HALTERNATE ACTION/3(50X,4I3/)///)
0379        15000 FORMAT(20X,17HPARAMETER UPDATES /20X,8HATTITUDE,5X,16I2/18X,
                 .10HNAVIGATION,5X,11I2///)
0380        16000 FORMAT(1H1)
0381              END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	9042	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	1149	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	976	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3	DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4	CONTRL	12	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5	ENVIR	404	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6	GFPART	800	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7	NOISE	216	PIC OVR REL GBL SHR NOEXE RD WRT LONG
8	TARGETS	36	PIC OVR REL GBL SHR NOEXE RD WRT LONG
9	ASTATE	160	PIC OVR REL GBL SHR NOEXE RD WRT LONG
10	FILTER	88	PIC OVR REL GBL SHR NOEXE RD WRT LONG
11	SEQU	292	PIC OVR REL GBL SHR NOEXE RD WRT LONG
12	TMAT	312	PIC OVR REL GBL SHR NOEXE RD WRT LONG
13	NSTATE	112	PIC OVR REL GBL SHR NOEXE RD WRT LONG
14	ARRAYS	1096	PIC OVR REL GBL SHR NOEXE RD WRT LONG
15	TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG
16	CONST	112	PIC OVR REL GBL SHR NOEXE RD WRT LONG
17	UPDT	752	PIC OVR REL GBL SHR NOEXE RD WRT LONG
18	PLOT	16	PIC OVR REL GBL SHR NOEXE RD WRT LONG
19	PHIA	7176	PIC OVR REL GBL SHR NOEXE RD WRT LONG
20	PHIN	1440	PIC OVR REL GBL SHR NOEXE RD WRT LONG
21	GCPDAT	192	PIC OVR REL GBL SHR NOEXE RD WRT LONG
22	STARPAR	560	PIC OVR REL GBL SHR NOEXE RD WRT LONG
23	COMPOSIT	19624	PIC OVR REL GBL SHR NOEXE RD WRT LONG
24	LMTPAR	392	PIC OVR REL GBL SHR NOEXE RD WRT LONG
25	MEASOUT	648	PIC OVR REL GBL SHR NOEXE RD WRT LONG
26	GPSPAR	192	PIC OVR REL GBL SHR NOEXE RD WRT LONG
27	PART	2112	PIC OVR REL GBL SHR NOEXE RD WRT LONG
28	CLOUD	96	PIC OVR REL GBL SHR NOEXE RD WRT LONG
29	TITLE	40	PIC OVR REL GBL SHR NOEXE RD WRT LONG
30	MODE	40	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		INDATA

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000280	R*8	A1	2-00000288	R*8	A2	2-00000290	R*8	A3	2-00000298	R*8	A4
2-000002A0	R*8	A5	2-000002A8	R*8	A6	2-000002B0	R*8	A7	2-000002B8	R*8	A8
24-00000000	R*8	AL	16-00000000	R*8	ATM	7-000000C8	R*8	BRE	2-000002C0	R*8	C1
2-000002C8	R*8	C2	2-000002F0	R*8	C3	2-000002D0	R*8	C3S1	15-00000038	R*8	DATE0
15-00000070	R*8	DATER	15-00000020	R*8	DEL	25-000001E0	R*8	DHCL	15-00000068	R*8	DTA
11-000000F0	R*8	DTGPS	15-00000030	R*8	DTN	15-00000080	R*8	DTPRINT	21-000000B8	R*8	DTPS
11-000000E8	R*8	DTST	16-00000060	R*8	DTU	25-000001E8	R*8	DVCL	25-00000278	R*8	EDHL
25-00000280	R*8	EDVL	21-000000B0	R*8	GCPN	2-00000300	I*4	I	2-00000308	I*4	ICOVDI
3-00000004	I*4	IDEBUG	21-000000AC	I*4	IDLMAX	3-00000000	I*4	IENTER	11-000000E4	L*4	IFLAG

2-0000030C	I*4	IGCP	5-00000190	I*4	INIT	8-00000004	I*4	IS	2-00000314	I*4	IST
2-00000304	I*4	J	16-00000048	R*8	J2	16-00000050	R*8	J3	16-00000058	R*8	J4
8-0000000C	L*4	JFLAG	2-00000310	I*4	K	24-00000010	R*8	LAT	21-000000A0	I*4	LINNUM
24-00000008	R*8	LON	8-00000010	I*4	MCODE	4-00000000	I*4	MOP	8-00000000	I*4	MTYPE
8-00000008	I*4	NS	8-00000014	R*8	PI	16-00000068	R*8	PK1	19-00001C00	R*8	QMAX
13-00000068	R*8	RADE	13-00000060	R*8	RADM	16-00000010	R*8	RBE	16-00000008	R*8	RBM
16-00000018	R*8	RBO	16-00000020	R*8	RE2	16-00000028	R*8	RM2	2-000002F8	R*8	S1
2-000002D8	R*8	S2	2-000002E0	R*8	S3	24-00000178	R*8	SIGGCP	7-000000D0	R*8	SRE
7-000000C0	R*8	SRM	21-000000A4	R*8	TGCP	24-00000180	R*8	THET	15-00000018	R*8	TIA
15-00000000	R*8	TIME	15-00000028	R*8	TIN	4-00000004	R*8	TINT	15-00000058	R*8	TIS
15-00000060	R*8	TISN	15-00000048	R*8	TMEAS	15-00000008	R*8	TNEXT	18-00000000	R*8	TP1
18-00000008	R*8	TP2	8-0000001C	R*8	TPI	15-00000078	R*8	TPRINT	15-00000050	R*8	TRACK
2-000002E8	R*8	TSTART	15-00000010	R*8	TSTOP	15-00000040	R*8	TZERO	16-00000040	R*8	UE
16-00000030	R*8	UM	16-00000038	R*8	US						

ARRAYS

Address	Type	Name	Bytes	Dimensions
12-00000000	R*8	A	72	(3, 3)
29-00000000	L*1	ATITLE	40	(40)
12-00000048	R*8	B	72	(3, 3)
7-00000060	R*8	BD	24	(3)
7-00000090	R*8	BDD	24	(3)
24-000000C8	R*8	BKL	16	(2)
24-000000A8	R*8	BL	16	(2)
22-00000000	R*8	BS	32	(2, 2)
7-00000030	R*8	BSF	24	(3)
22-00000160	R*8	BSK	32	(2, 2)
7-00000000	R*8	BWD	24	(3)
12-00000090	R*8	C	72	(3, 3)
28-00000000	R*8	CLOTBL	96	(12)
23-00001E40	R*8	COV	3872	(22, 22)
19-00000C00	R*8	COVA	2048	(16, 16)
20-00000360	R*8	COVN	288	(6, 6)
9-00000070	R*8	D	24	(3)
9-00000088	R*8	DD	24	(3)
9-00000000	R*8	DE	32	(4)
25-00000150	R*8	DHCS	16	(2)
25-00000160	R*8	DVCS	16	(2)
9-00000020	R*8	E	32	(4)
6-00000080	R*8	EA	384	(4, 12)
25-00000258	R*8	EDHS	16	(2)
25-00000268	R*8	EDVS	16	(2)
12-00000008	R*8	EM	96	(4, 3)
25-00000228	R*8	EMXG	48	(6)
6-00000000	R*8	FA	128	(4, 4)
6-00000200	R*8	FN	288	(6, 6)
11-00000084	I*4	IAA	48	(3, 4)
11-000000F8	I*4	IDELAY	44	(11)
11-00000000	I*4	IES	100	(25)
21-00000000	I*4	IGCPDT	160	(4, 10)
23-00003C80	I*4	IP	88	(22)
10-00000018	I*4	IPA	64	(16)
10-00000000	I*4	IPN	24	(6)
11-00000064	I*4	ITS	80	(4, 5)

25-000001F0	R*8	LMU	24	(3)
30-00000000	I*4	MODE	40	(10)
25-00000170	R*8	MS	48	(3, 2)
25-00000000	R*8	MX	48	(6)
23-00003088	R*8	P	3872	(22, 22)
19-00000000	R*8	PA	128	(4, 4)
26-00000000	R*8	PB	24	(3)
26-00000060	R*8	PBK	24	(3)
19-00000200	R*8	PDA	512	(4, 16)
27-000006E0	R*8	PDHL	176	(22)
27-00000420	R*8	PDHS	352	(22, 2)
20-00000120	R*8	PDN	288	(6, 6)
27-00000730	R*8	PDVL	176	(22)
27-00000530	R*8	PDVS	352	(22, 2)
23-00000000	R*8	PHI	3872	(22, 22)
19-00000400	R*8	PHIA	2048	(16, 16)
20-00000240	R*8	PHIN	288	(6, 6)
20-00000000	R*8	PN	288	(6, 6)
23-00002060	R*8	PO	3872	(22, 22)
19-00001400	R*8	POA	2048	(16, 16)
20-00000480	R*8	PON	288	(6, 6)
2-00000108	R*8	PQ	128	(4, 4)
5-00000050	R*8	PROFILE	320	(10, 4)
26-00000030	R*8	PS	24	(3)
26-00000090	R*8	PSK	24	(3)
2-000000C0	R*8	PTP	72	(3, 3)
27-00000000	R*8	PX	176	(22)
27-00000210	R*8	PXD	176	(22)
27-00000080	R*8	PY	176	(22)
27-000002C0	R*8	PYD	176	(22)
27-00000160	R*8	PZ	176	(22)
27-00000370	R*8	PZD	176	(22)
17-00000080	R*8	Q	288	(6, 6)
17-00000030	R*8	QA	128	(16)
17-00000100	R*8	QDOT	288	(6, 6)
17-00000000	R*8	QN	48	(6)
23-00000F20	R*8	QT	3872	(22, 22)
25-00000030	R*8	RGPS	288	(6, 6)
25-00000208	R*8	RL	32	(2, 2)
25-000001A0	R*8	RS	64	(2, 2, 2)
7-00000078	R*8	SD	24	(3)
2-000001E8	R*8	SDA	128	(16)
7-000000A8	R*8	SDD	24	(3)
2-00000268	R*4	SDN	24	(6)
9-00000058	R*8	SF	24	(3)
24-00000098	R*8	SKL	16	(2)
24-00000088	R*8	SL	16	(2)
22-00000020	R*8	SS	32	(2, 2)
7-00000048	R*8	SSF	24	(3)
22-00000180	R*8	SSK	32	(2, 2)
5-00000000	R*8	STATE	80	(10)
7-00000018	R*8	SWD	24	(3)
14-00000000	R*8	T1	24	(3)
14-00000098	R*8	T11	72	(3, 3)
14-00000018	R*8	T2	24	(3)
14-00000030	R*8	T3	24	(3)
14-000000E0	R*8	T33	72	(3, 3)

```
14-00000048   R*8   T4          80   (10)
14-0000012B   R*8   T44        128   (4, 4)
14-000003E8   R*8   T5          32   (4)
14-00000408   R*8   T6          32   (4)
14-000001A8   R*8   T66        288   (6, 6)
14-00000428   R*8   T7          32   (4)
14-000002C8   R*8   T77        288   (6, 6)
19-00000080   R*8   TA         384   (4, 12)
24-00000018   R*8   TBNL        72   (3, 3)
22-00000000   R*8   TBNS       144   (3, 3, 2)
24-00000130   R*8   TIEO        72   (3, 3)
 2-000001B8   R*8   TMP         96   (3, 4)
24-00000060   R*8   TNL         72   (3, 3)
24-000000E8   R*8   TNLK        72   (3, 3)
22-00000040   R*8   TNS        144   (3, 3, 2)
22-000001A0   R*8   TNSK       144   (3, 3, 2)
 2-00000000   R*8   TP          96   (4, 3)
 2-00000060   R*8   TPT         96   (3, 4)
26-00000018   R*8   VB          24   (3)
26-00000078   R*8   VBK         24   (3)
26-00000048   R*8   VS          24   (3)
26-000000A8   R*8   VSK         24   (3)
 9-00000040   R*8   WJ          24   (3)
13-00000030   R*8   X           48   (6)
13-00000000   R*8   XD          48   (6)
23-00003C08   R*8   XT         176   (22)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000000	4'	1-00000007	10'	1-0000000D	20'	1-00000019	30'	1-0000002B	40'	1-00000043	45'
1-0000004B	60'	1-00000053	70'	1-0000005B	80'	1-00000063	90'	**	100'	1-0000006B	110'
1-00000073	115'	1-0000007B	120'	1-00000081	130'	1-00000085	140'	**	150'	1-0000008E	160'
1-00000095	165'	1-0000009C	166'	1-000000A3	170'	1-000000AA	180'	1-000000B1	190'	1-000000B8	195'
1-000000BF	200'	1-000000C9	210'	**	400	**	450	**	500	0-00001E37	1000
**	1100	**	2000	**	2100	**	2200	0-00001F22	2500	**	3000
**	3100	**	3300	**	4000	**	5000	**	6000	**	7000
**	7500	0-00002008	8000	1-000000CE	9000'	1-00000181	10000'	1-000001AC	11000'	1-000001DA	11200'
1-000001F7	11300'	1-00000230	11400'	1-0000026D	11500'	1-0000029F	11600'	1-000002D5	12000'	1-00000344	13000'
1-0000039E	14000'	1-00000430	15000'	1-00000471	16000						

FUNCTIONS AND SUBROUTINES REFERENCED

FOR$OPEN   MATAB   MTH$DCOS   MTH$DSIN

Total Space Allocated = 48231 Bytes

```
COMMAND QUALIFIERS

  FORTRAN /L INDATA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:            22.24 seconds
  Elapsed Time:        68.35 seconds
  Page Faults:         1281
  Dynamic Memory:      298 pages
```

2.2 Event Sequencer (GCPSEQ)

One of the primary functions of GCPSIM is to analyze the effect of different measurement sequences and to establish a baseline sequence which will meet the operational requirements. The operations sequencer (GCPSEQ) has been designed to allow any sequence of operations to be simulated.

The GCPSEQ module is responsible for controlling the type of measurements performed by the onboard GCP detection system. The possible measurement types are identified by an event code as illustrated in Table A-1. The delay times listed in the table were chosen to be multiples of the cycle time between two consecutive scans of the mirror in the MSS. This would allow the implementation of a scan initialization signal which would be generated by the onboard navigation system to control the scanner of the MSS. Control of the scan mechanism in this manner would eliminate the distortions (as large as 28 pixels) associated with variation of the scan period and would reduce the amount of onboard processing required for image correction. The primary interrupt period of the onboard navigation system will be chosen such that it divides evenly into the scan period.

An event sequencing table has been established to identify series of events to be performed. The overall mission may be broken down into different types of operational sequences defined by partitioning locations in the event sequencing table containing the event codes to be performed. A sample event sequence table is shown in Table A-1. In the table, the first partition contains a sequence which performs a GPS measurement followed by a 10 second wait. The second partition contains a sequence where a GPS measurement is followed by a #1 star tracker meaurement, a 10 second wait, a second GPS measurement, and a #2 star tracker measurement.

PARTITION NUMBER	1		2					3								4			
SEQUENCE NUMBER	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
EVENT CODES	2	8	2	3	8	2	4	2	6	3	8	2	6	4	8	3	6	4	6

Table A-1. Event Sequence Table

Partitions within the event sequence table are defined by an event cycling table. The cycling table defines partitions in terms of the sequence numbers which mark the beginning and end of the sequence within the partition.

In addition to defining partitions the event cycling table defines either the number of repetitions of a partition or the maximum amount of time to be spent cycling through a partition. Operations within a partition are terminated when one of these limits is reached. A sample event cycling table is shown in Table A-2. A partition is defined in terms of the event sequence numbers. In the table, the first event group defines a partition beginning at sequence number 1 and ending with sequence number 2. Operations within the partition will be repeated five times or for two hours depending on which occurs first.

PARTITION NUMBER	SEQUENCE START	SEQUENCE STOP	CYCLE REP. LIMIT	CYCLE TIME (HOURS)
1	1	2	5	2
2	3	7	100	15
3	1	2	50	2
4				
5				

Table A-2. Sample Event Cycling Table

GCP measurement are considered to take precedence over all other types of events and will therefore be scheduled on a separate basis. A GCP count table (Table II-IV) containing all information necessary to locate the predicted GCP locator has been implemented in the sequencer.

The location and size of the search area is completely defined by the line start, line stop, element start, and element stop indicators contained in the count table as shown in Figure A-5 .

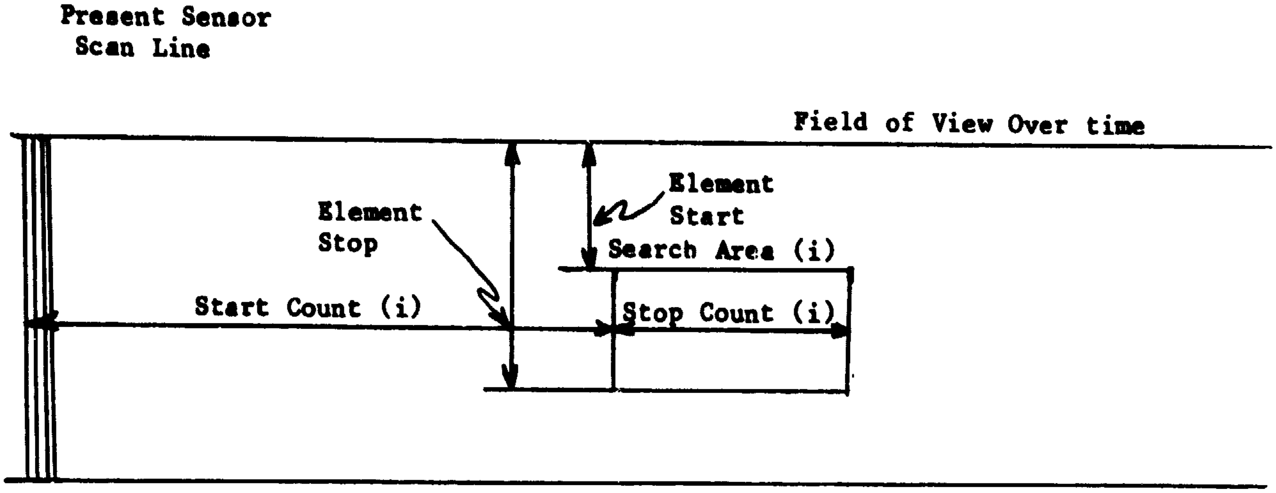


Figure A-5 Explanation of Parameters in Count Table

In Figure A-5 the two long horizontal lines enclose a diagram of the imaging sensor's field of view over time. The vertical swaths depict one scan of the sensor's instantaneous field of view, and the small squares within the scan line are individual picture elements. The start count contains the number of scans between the present scan line and the first line of the search area. The stop count contains the size of the search area in scan lines. Once it is known that the search area is within the field of view, it is necessary to determine its location. The element start count contains the distance, in picture elements, of the search area from the start of the scan line. The element stop count contains the distance to the end of the search area from the start of the scan line. These four parameters totally describe the position and size of the search area relative to the current position of the sensor's field of view. The center of the search area is coincident with the estimated center of the landmark and its size is a direct function of the uncertainty in satellite position and attitude.

The functional flow of the sequencer is illustrated by the VCLR in Figure A-6. On the first entry into the sequencer, all indices in the sequence table, the cycle repetition table and the GCP count table are initialized. On all other entries to the sequencer, the event sequence index, ISS, is incremented. A test is conducted to determine if the sequence index has exceeded the bounds established by the cycling table. If these bounds have not been exceeded the event code is read from the sequence table. If the limits have been exceeded, the sequence index is set back to the start of the event partition defined by the first column in the cycling table. The cycle repetition index is then incremented to indicate the total number of cycles through that partition. If the number of cycles does not exceed the limits established by the cycling table the event code is obtained directly from the sequence table. If the total number of cycles exceeds the limits, the cycling rep index and the cycle time are reinitialized to zero and the next partition entered by incrementing the event group counters. If the event group counter exceeds the number of partitions set up in the cycling table, the simulation is effectivly terminated.

Following extraction of the event code, MCODE, from the sequence table a test is made to determine if the event is a measurement or a delay. If MCODE is greater than five, indicating a delay, the measurement time is found by adding the length of the delay to the current time.

If MCODE denotes a measurement a sequence test is initiated. First, a test is made to determine if a GPS update is to be made. If it is then the measurement time is found by

$$TMEAS = TIME+DTGPS$$

where: DTGPS is the time required to make the GPS measurement

If the measurement is not a GPS update a test is made to determine if a measurement is being made by star tracker one. If this is the case, a vector from the star tracker to the star is generated by the subroutine BVECT. BVECT is described later in this chapter. Since the star being observed by the tracker might be obscured by a major body such as the sun, moon, or earth, a test is made by the subroutine VISIBLE to determine if the tracker is occulted. VISIBLE is described later in this chapter. If the first star tracker is occulted the process is repeated for the second star tracker, and if both trackers are occulted the sequencer will delay measurements for a period of time. If the measurement code originally indicated a star tracker two sighting, the process is performed in the reverse sequence. Assuming that one of the star trackers is not occulted the measurement time is found by

$$TMEAS = TIME+ DTST$$

where: DTST is the time required to make a star tracker measurement

Following computation of the measurement time, either through delay sequencing or measurement sequencing, a test is performed to determine if there is sufficient time prior to the next GCP sighting for the measurement to be made. First, the number of scan lines to the GCP is computed by

LINLEFT = LINNUM-(TIME-TGCP)/DTPS

were: LINLEFT = number of scan lines remaining to the GCP

LINNUM = number of times between last GCP and next GCT
TIME = current time
TGCP = the time at which last GCP was sighted
DTPS = time required per scan line

If the number of scan lines left to the GCP is larger than the number of lines that will be scanned during the planned measurement, the measurement is made. If there is insufficient time remaining, the measurement code is set to 1 indicating a GCP sighting and TMEAS is computed.

Sequence Operations

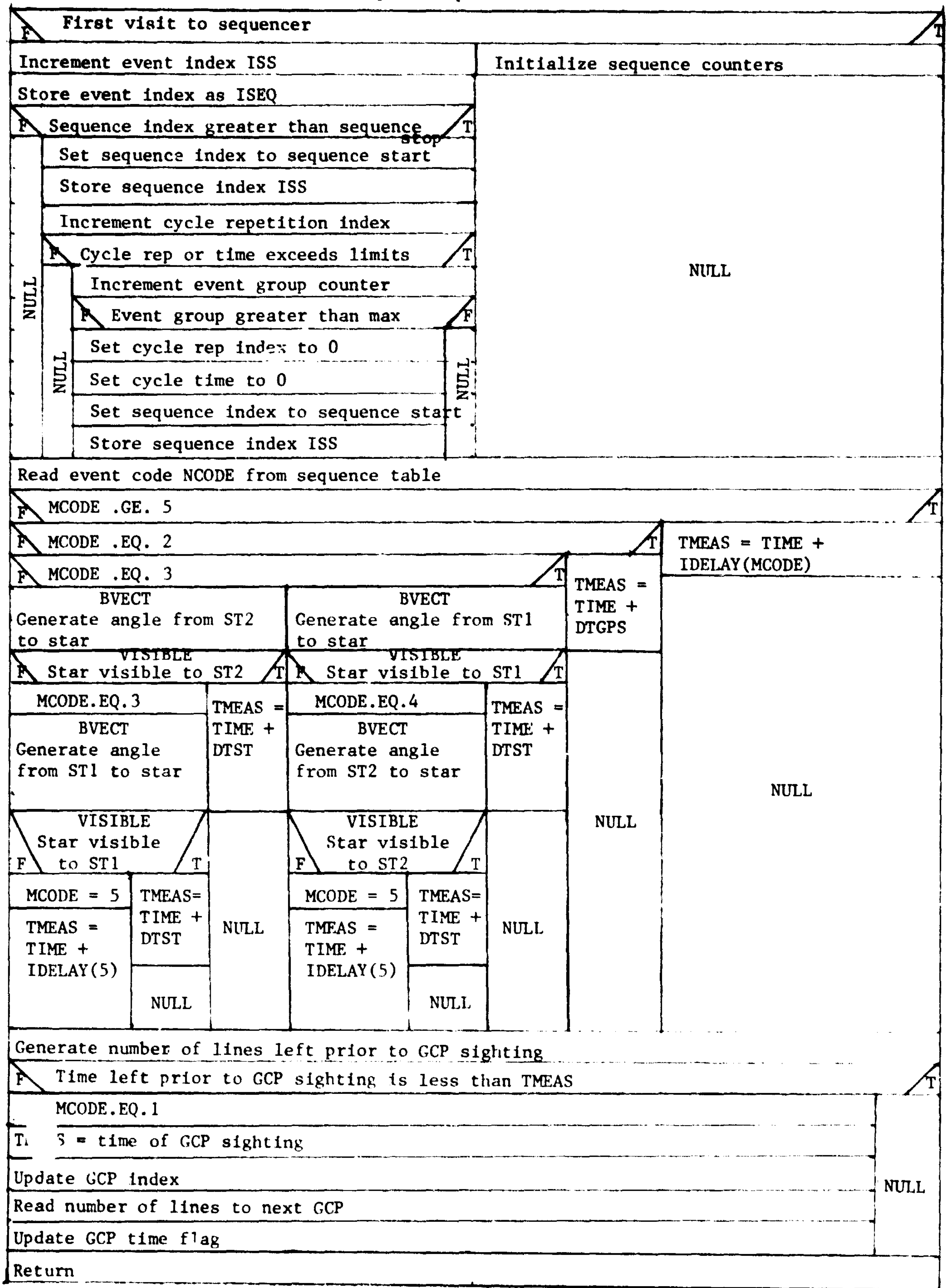


Figure A-6 GCPSEQ VCLR

```
0001          SUBROUTINE  GCPSEQ
0002          REAL*8 ANGLE,CT,ST,VEC
0003          INCLUDE 'DEBUG.COM'
0004  *       COMMON /DEBUG/ IENTER,IDEBUG
      * C
      * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
      * C
      * C                    IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
      * C                    IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
      * C
0005          INCLUDE 'TARGETS.COM'
0006  *       COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
0007  *       LOGICAL JFLAG
0008  *       REAL*8 PI,TPI
      * C
      * C              MEASUREMENT SPECIFICATIONS
      * C
      * C                    MTYPE     MEASUREMENT TYPE
      * C                    JFLAG     SET FOR STAR OBSTRUCTION
      * C                    MCODE      "    "  MEASUREMENT PROCESSING
      * C                    PI        PI
      * C                    TPI       2*PI
      * C
0009          INCLUDE 'SEQU.COM'
0010  *       COMMON /SEQU/  IS(25),ITS(4,5),IAA(3,4),IFLAG
      *      .               ,DTST,DTGPS,IDELAY(11)
0011  *       REAL*8 DTST,DTGPS
0012  *       LOGICAL IFLAG
      * C
      * C              SEQUENCE OF EVENTS TABLES
      * C
      * C                    IS        EVENT SCHEDULAR
      * C                    ITS       CYCLE SEQUENCER
      * C                    IAA       ALTERNATE ACTIONS
      * C                    IFLAG     SET WHEN TARGET IS OBSTRUCTED
      * C
0013          INCLUDE 'TIME.COM'
      * C
0014  *       COMMON /TIME/  TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
      *      .               ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0015  *       REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
      *      .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
      * C
      * C              THESE ARE THE TIME REFERENCE FRAMES
      * C
      * C                    TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
      * C                    TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
      * C                    TSTOP     RUN TERMINATION TIME (SEC)
      * C                    TIA       ATTITUDE INTEGRATION TIME (SEC)
      * C                    DEL         "         "        STEP SIZE (SEC)
      * C                    TIN       POSITION INTEGRATION TIME (SEC)
      * C                    DTN         "         "        STEP SIZE (SEC)
      * C                    DATEO     DATE OF FLIGHT EPOCH (JD)
      * C                    DATER     DATE OF 1950 EPOCH (JD)
      * C                    TZERO     START TIME IN SECS. SINCE DATEO
      * C                    TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
      * C                    TIS       REAL WORLD REFERENCE TIME (SEC)
      * C
```

```
 * C                      TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
 * C                      DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
 * C                                TOO LARGE AT MEASUREMENT TIME
 * C                      TPRINT    TIME FOR PRINT (SEC)
 * C                      DTPRINT   INCREMENT ON TPRINT (SEC)
           INCLUDE 'GCPDAT.COM'
 *           COMMON /GCPDAT/ IGCPDT(4,10),LINNUM,TGCP,IDLMAX,GCPN,DTPS
 *           REAL*8 TGCP,GCPN,DTPS
           DIMENSION MT(15),MS(15),VEC(3)
   C
   C*************************************************************************
   C          THIS PROGRAM DETERMINES EVENT SEQUENCING
   C
   C          THE PROGRAM PARAMETERS ARE:
   C                TIME     THE CURRENT FLIGHT TIME
   C                MTYPE    THE TYPE OF MEASUREMENT
   C                     MTYPE = 1 FOR GCP MEASUREMENTS
   C                     MTYPE = 2 FOR GPS NAVIGATION
   C                     MTYPE = 3 FOR STAR TRACKER 1 MEASUREMENTS
   C                     MTYPE = 4 FOR STAR TRACKER 2 MEASUREMENTS
   C                IS       THE STAR NUMBER SPECIFICATION
   C                JFLAG    OPERATIONAL FLAG - SET IF THE
   C                            TARGET IS NOT AVAILABLE
   C
   C          THE LABELED COMMON (SEQ) PARAMETERS ARE:
   C                IES EVENT SCHEDULE TABLE
   C                ITS TIME CYCLING TABLE
   C                IDT STAR SPECIFICATION TABLE
   C          MT AND MS RELATE MTYPE AND MSET TO MCODE, RESPECTIVELY
   C
   C         MCODE        ACTION
   C
   C           1            GCP MEASUREMENT
   C           2            GPS NAVIGATION
   C           3            STAR TRACKER 1 MEASUREMENT
   C           4            STAR TRACKER 2 MEASUREMENTS
   C           5            DELAY OF XXXXXX US
   C           6            DELAY OF XXXXXX US
   C           7            DELAY OF XXXXXX US
   C           8            DELAY OF XXXXXX US
   C           9            DELAY OF XXXXXX US
   C          10            DELAY OF XXXXXX US
   C          11            DELAY OF XXXXXX US
   C          12            DELAY OF XXXXXX US
   C          13            DELAY OF XXXXXX US
   C          14            DELAY OF XXXXXX US
   C          15            DELAY OF XXXXXX US
   C
   C*************************************************************************
   C
            IF(IENTER.EQ.1)WRITE(6,999)
   999   FORMAT(' ENTERING GCPSEQ ')
   C
   C*************************************************************************
   C         FLAG.EQ. FALS_ INDICATES NEW SEQUENCE TABLES
   C*************************************************************************
```

```
          C
0022              IF(IFLAG) GO TO 10
0023              ISEQ  = ITS(1,1)
0024              ISS   = ISEQ
0025              IR    = 1
0026              IC    = 1
0027              LINNUM= IGCPDT(1,1)
0028              TGCP  = TIME
0029              GCPN  = 1
0030              IFLAG = .TRUE.
0031              GO TO 40
          C
          C*****************************************************************
          C       INCREMENT SEQUENCE
          C*****************************************************************
          C
0032      10    ISS = ISS + 1
0033            ISEQ = ISS
0034      20    IF(ISEQ.LE.ITS(2,IC)) GO TO 40      ! LIMIT OF PARTITION
0035      30    ISEQ = ITS(1,IC)                    ! SET SEQUENCE TO START
0036              ISS  = ISEQ
0037              IR   = IR + 1
0038              IF(IR.LT.ITS(3,IC)) GO TO 40
          C                                         ! PARTITION REPITITIONS GT MAX
0039              IC  = IC + 1                      ! INCREMENT PARTITION NUMBER
0040              IF(IC.GT.IPMAX)GO TO 80           ! PARTITION NUMBER GT MAX
0041              IR   = 0                          ! REINITIALIZE
0042              ST   = TIME
0043              ISEQ = ITS(1,IC)
0044              ISS  = ISEQ
          C
          C*********************************************************
          C       GET MEASUREMENT CODE
          C*********************************************************
          C
0045      40      MCODE = IES(ISEQ)                  ! GET MEASUREMENT CODE
0046              IF(MCODE .NE. 1) GO TO 42
0047              WRITE(6,41)
0048      41      FORMAT(5X,'MCODE ERROR - MCODE =1')
0049              CALL EXIT
0050      42      IF(MCODE.GE.5) GO TO 85
          C
          C*****************************************************************
          C       GPS MEASUREMENTS?
          C*****************************************************************
          C
0051              IF(MCODE.NE.2)GO TO 45
0052              TMEAS = TIME + DTGPS
0053              GO TO 100
          C
          C*****************************************************************
          C        STAR TRACKER 1 OR 2?
          C*****************************************************************
          C
0054       45      IF(MCODE.EQ.3) GO TO 50
          C
          C*****************************************************************
```

```
C         STAR TRACKER 2: CHECK TO SEE IF VISIBLE AND GET ANGLE TO STAR
C**************************************************************************
C
                CALL BVECT(VEC,2)
                CALL VISIBLE(VEC,4,JFLAG)        ! IS TRACKER OCCULTED
                IF(.NOT. JFLAG) GO TO 55
C
C**************************************************************************
C         OTHERWISE STAR TRACKER 1 MEASUREMENTS
C**************************************************************************
C
                MCODE=3
                CALL BVECT(VEC,1)
                CALL VISIBLE(VEC,4,JFLAG)        ! IS TRACKER OCCULTED
                IF(.NOT. JFLAG) GO TO 55
                GO TO 85
C
C**************************************************************************
C          STAR TRACKER 1 MEASUREMENTS
C**************************************************************************
C
50              CALL BVECT(VEC,1)
                CALL VISIBLE(VEC,4,JFLAG)        ! IS TRACKER OCCULTED
                IF(.NOT. JFLAG) GO TO 55
C
C**************************************************************************
C          OTHERWISE STAR TRACKER 2
C**************************************************************************
C
                MCODE=4
                CALL BVECT(VEC,2)
                CALL VISIBLE(VEC,4,JFLAG)        ! IS TRACKER OCCULTED
                IF(.NOT. JFLAG) GO TO 55
                GO TO 85
 55             TMEAS = TIME + DTST
                GO TO 100
C
C********************************************************
C               JUMP TO LAST DELAY
C********************************************************
C
80              MCODE=15
                GO TO 87
C
C********************************************************
C               SET UP NORMAL DELAY
C********************************************************
C
85              MCODE=5
87              TMEAS=TIME+IDLAY(MCODE-4)
C
C*****************************************************************
C               CHECK FOR TIM  TO MAKE GCP MEASUREMENTS
C*****************************************************************
C
100             LINLEFT=LINNUM-(TIME-TGCP)/DTPS
                IF(LINLEFT .GT. (TMEAS-TIME)/DTPS) GO TO 200
```

```
      C
      C*****************************************************************
      C         GCP MEASUREMENTS
      C*****************************************************************
      C
   60         MCODE=1
              TMEAS = (TIME + LINLEFT*DTPS)
      C                                   ! COMPUTE MEASUREMENT TIME
      C       GCPN  = GCPN + 1       !TEMPORARY
              GCPN = 1               !TEMPORARY FOR LONG RUN - JACK
              LINNUM= IGCPDT(1,GCPN)
              TGCP  = TMEAS + IGCPDT(2,GCPN)/DTPS
  200         RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	553	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	59	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	236	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 TARGETS	36	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 SEQU	292	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7 GCPDAT	192	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GCPSEQ

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000090	R*8	ANGLE	2-00000098	R*8	CT	6-00000038	R*8	DATEO	6-00000070	R*8	DATER
6-00000020	R*8	DEL	6-00000068	R*8	DTA	5-000000F0	R*8	DTGPS	6-00000030	R*8	DTN
6-00000080	R*8	DTPRINT	7-000000B8	R*8	DTPS	5-000000E8	R*8	DTST	7-000000B0	R*8	GCPN
2-000000BC	I*4	I	2-000000B4	I*4	IC	3-00000004	I*4	IDEBUG	7-000000AC	I*4	IDLMAX
3-00000000	I*4	IENTER	5-000000E4	L*4	IFLAG	2-000000B8	I*4	IPMAX	2-000000B0	I*4	IR
4-00000004	I*4	IS	2-000000A8	I*4	ISEQ	2-000000AC	I*4	ISS	4-0000000C	L*4	JFLAG
2-000000C0	I*4	LINLEFT	7-000000A0	I*4	LINNUM	4-00000010	I*4	MCODE	4-00000000	I*4	MTYPE
4-00000008	I*4	NS	4-00000014	R*8	PI	2-000000A0	R*8	ST	7-000000A4	R*8	TGCP
6-00000018	R*8	TIA	6-00000000	R*8	TIME	6-00000028	R*8	TIN	6-00000058	R*8	TIS
6-00000060	R*8	TISN	6-00000048	R*8	TMEAS	6-00000008	R*8	TNEXT	4-0000001C	R*8	TPI
6-00000078	R*8	TPRINT	6-00000050	R*8	TRACK	6-00000010	R*8	TSTOP	6-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
5-000000B4	I*4	IAA	48	(3, 4)
5-000000F8	I*4	IDELAY	44	(11)
5-00000000	I*4	IES	100	(25)
7-00000000	I*4	IGCPCT	160	(4, 10)
5-00000064	I*4	ITS	80	(4, 5)
2-00000054	I*4	MS	60	(15)
2-00000018	I*4	MT	60	(15)
2-00000000	R*8	VEC	24	(3)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-0000006A	10	**	20	**	30	0-000000C1	40	1-00000014	41'	0-000000F8	42
0-0000011A	45	0-0000014E	50	0-0000017C	55	**	60	0-0000018E	80	0-00000197	85
0-0000019E	87	0-00000189	100	0-00000228	200	1-00000000	999'				

FUNCTIONS AND SUBROUTINES REFERENCED

BVECT FOR$EXIT VISIBLE

Total Space Allocated = 1512 Bytes

A-11

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time:	2.52 seconds
Elapsed Time:	27.61 seconds
Page Faults:	372
Dynamic Memory:	160 pages

2.2.1 CHECK FOR STAR OCCULTATION (VISIBLE)

Subroutine VISIBLE acts as an executive for the determination of whether the target star is occulted by a major body. The bodies checked are the earth, moon, and sun. The occultation geometry is checked by passing to subroutine OCCULT a unit vector to the target star, a vector to the center of the body of concern and the effective radius of the body of concern. OCCULT then returns a .TRUE. in the logic variable L if the star is occulted by the body. Figure A-7 is a VCLR of the process.

Input Variables and Output Variables

- VEC = Unit vector to the target star. (Unitless)
- RSO = Radius vector to the sun (KM)
- RBO = Obstruction radius of the sun (KM)
- RSM = Radius vector to the moon (KM)
- RBM = Obstruction radius of the moon. (KM)
- R = Radius vector to the earth. (KM)
- RBE = Obstructing radius of the earth (KM)
- L = Logical flag . TRUE. if occulted.

Calling Routines - GCPSEQ, MEASURE

Routine and Functions Called - OCCULT

VISIBLE - VCLR

Occulted by the sun?
Occulted by the moon?
Occulted by the earth?

Figure A-7

```
            SUBROUTINE VISIBLE(VEC,NO,L)
            DIMENSION VEC(3)
            REAL*8 V,ZETL,V C,VMAG
            LOGICAL L
            INCLUDE 'DEBUG.COM'
  *         COMMON /DEBUG/ IENTER,IDEBUG
  * C
  * C                USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
  * C
  * C                      I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
  * C                      IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
  * C
            INCLUDE 'RVEC.COM'
  *         COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
  *        .              ,RA,R2,R3,RSMA,RTG(3)
  *         REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
  * C
  * C                THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
  * C
  * C                      R          EARTH CENTER TO S/C -    ECI (KM)
  * C                      RM           "      "        MOON - ECI (KM)
  * C                      RO           "      "        SUN -  ECI (KM)
  * C                      RSM        SPACECRAFT TO MOON -        ECI (KM)
  * C                      RSO           "              SUN -        ECI (KM)
  * C                      RSS        EARTH CENTER TO STAR -   ECI
  * C                      RA         ABSOLUTE OF VECTOR R (KM)
  * C                      R2         SQUARE OF RA (KM 2)
  * C                      R3         CUBE OF RA (KM 3)
  * C                      RSMA       ABSOLUTE OF RSM (KM)
  * C
            INCLUDE 'CONST.COM'
  *         COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
  *         REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
  * C
  * C                PROGRAM CONSTANTS
  * C
  * C                      ATM        S/C AREA TO MASS RATIO (METERS/KG)
  * C                      RBM        OBSTRUCTION RADIUS OF THE MOON (KM)
  * C                      RBE           "           "           EARTH (KM)
  * C                      RBO           "           "           SUN (KM)
  * C                      R.2        SQUARE OF THE EARTHS RADIUS (KM 2)
  * C                      RM2          "              LUNAR RADIUS (KM 2)
  * C                      UM         LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
  * C                      US         SOLAR         "            "          "
  * C                      U.         EARTH         "            "          "
  * C                      J2,J3,J4   ZONAL GRAVITATIONAL HARMONIC TERMS
  * C                      DTU        REGULARIZED TIME STEP SIZE (SEC)
  * C                      PKI        SOLAR PRESSURE CONSTANT
  * C
    C
    C
    C           STAR AVAILABILITY
    C
    C
            IF(IENTER.EQ.1) WRITE(6,999) VEC,NO
      999   FORMAT(' ENTERING VISIBLE ',3F15.5,I5)
            IF(NO.EQ.1) GO TO 10
```

```
              C
              C************************************************************************
              C       IS THE TARGET OCCULTED BY THE SUN
              C************************************************************************
     0016           CALL OCCULT(RSO,VEC,RBO,L)
     0017           IF(L) GO TO 40
     0018           IF(NO.EQ.2) GO TO 20
              C************************************************************************
              C       IS THE TARGET OCCULTED BY THE MOON
              C************************************************************************
     0019     10    CALL OCCULT(RSM,VEC,RBM,L)
     0020           IF(L) GO TO 40
     0021           IF(NO.EQ.3) GO TO 30
              C************************************************************************
              C       IS THE TARGET OCCULTED BY THE EARTH
              C************************************************************************
     0022     20    CALL OCCULT(R,V C,RBE,L)
     0023           IF(L) GO TO 40
     0024           IF(NO.EQ.4) GO TO 50
              C************************************************************************
              C       IS THE TARGET OCCULTED BY THE S/C
              C************************************************************************
     0025     30    V = VMAG(VEC,3)
     0026           IF(VEC(3)/V.GT.ZETL) GO TO 50
     0027     40    L = .TRUE.
     0028     50    RETURN
     0029           END
```


PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	197	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	32	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	108	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	RVEC	224	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	CONST	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		VISIBLE

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-00000000	R*8	ATM	5-00000060	R*8	DTU	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER
5-00000048	R*8	J2	5-00000050	R*8	J3	5-00000058	R*8	J4	AP-0000000C@	L*4	L
AP-00000008@	I*4	NO	5-00000068	R*8	P[illegible]	4-00000080	R*8	R2	4-00000088	R*8	R3

4-000000A8	R*8	RA	5-00000010	R*8	RBE	5-00000008	R*8	RBM	5-00000018	R*8	RBO
5-0000002	R*8	RE2	5-00000028	R*8	RM2	4-000000C0	R*8	RSMA	5-00000040	R*8	UE
5-00000030	R*8	UM	5-00000038	R*8	US	2-00000000	R*8	V	2-00000008	R*8	ZETL

ARRAYS

Address	Type	Name	Bytes	Dimensions
4-00000000	R*8	R	24	(3)
4-00000018	R*8	RM	24	(3)
4-00000030	R*8	RO	24	(3)
4-00000048	R*8	RSM	24	(3)
4-00000060	R*8	RSO	24	(3)
4-00000078	R*8	RSS	24	(3)
4-000000C8	R*8	RTG	24	(3)
4-00000090	R*8	SB	24	(3)
AP-00000004@	R*8	VEC	24	(3)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-00000069	10	0-00000085	20	0-000000A1	30	0-000000C0	40	0-000000C4	50	1-00000000	999'

FUNCTIONS AND SUBROUTINES REFERENCED

OCCULT VMAG

Total Space Allocated = 681 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 1.19 seconds
Elapsed Time: 13.87 seconds
Page Faults: 351
Dynamic Memory: 160 pages

2.2.1.1 SUBROUTINE OCCULT (RSB, RSS, RB,L)

Subroutine OCCULT determines whether the target star is occulted by the major body of concern.

Processing Requirements

The processing uses the dot product of the two vectors RSS and RSB and compares the sine of the included angle with the sine of the angle opposite RB in such a manner as to not require trigonometric functions.

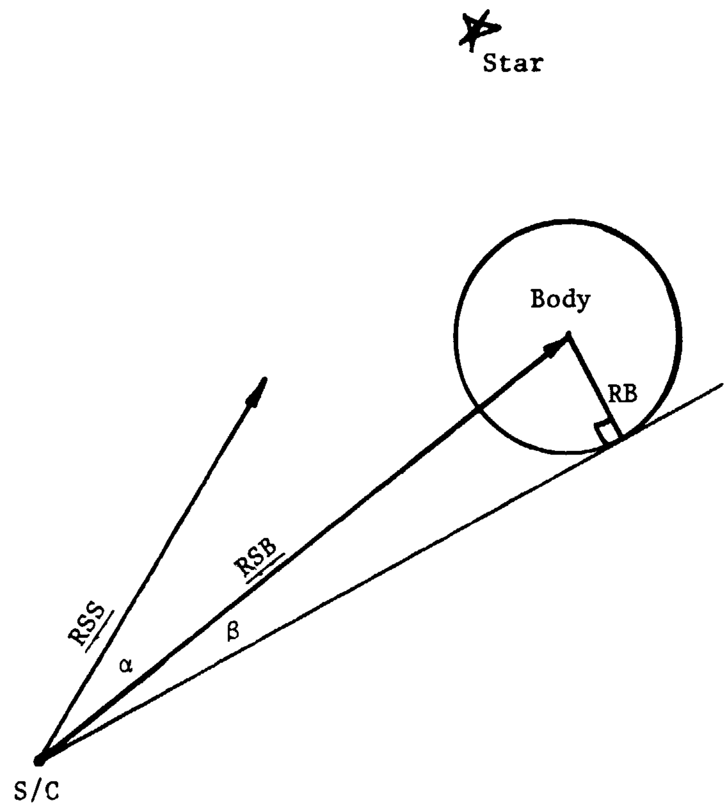


Figure A-8 Occultation Geometry

In referring to Figure A-8, if α is greater than β, the star will not be occulted. Further it may be seen that $\beta < 90$ for all spherical bodies. Therefore, given

$$\underline{RSS} \cdot \underline{RSB} = |\underline{RSS}|\,|\underline{RSB}| \cos \alpha$$

if the following condition does not exist the star is occulted

Test A

$$\underline{RSS} \cdot \underline{RSB} \leq 0 \ , \ \alpha \geq 90^0$$

A second condition that must exists is that be greater than β.

$\sin\alpha > \sin\beta$ is also a valid condition indicating $\alpha > \beta$.

and so is

$(\sin\alpha)^2 > (\sin\alpha)^2$ provides $\alpha < 90^0$

Since

$$(\sin\alpha)^2 = 1 - (\cos\alpha)^2 = 1 - \left(\frac{\underline{RSS} \cdot \underline{RSB}}{|\underline{RSS}|\,|\underline{RSB}|}\right)$$

and

$$(\sin\beta)^2 = \left(\frac{RB}{|\underline{RSB}|}\right)^2$$

Non occuitation is therefore implied by

$$1 - \left(\frac{\underline{RSS} \cdot \underline{RSB}}{|\underline{RSS}|\,|\underline{RSB}|}\right)^2 > \left(\frac{RB}{|\underline{RSB}|}\right)^2$$

or Test B

$$|\underline{RSB}|^2 - \left(\frac{\underline{RSS} \cdot \underline{RSB}}{|\underline{RSS}|}\right)^2 > (RB)^2$$

Both of these expressions (A) and (B) are used in the program. Figure A-9 is a VCLR of the process.

Input Requirements

RSS = BSI unit vector to the target star (unitless)
RSB = BCI vector to the body of concern. (KM)
RB = Obstruction radius of the body of concern. (KM)

Output Requirements

L - Logical variable. TRUE. if the target star is occulted, otherwise .FALSE.

Subroutine and Functions Called - VMAG, VDOT
Calling Subroutine - VISIBLE

OCCULT - VCLR

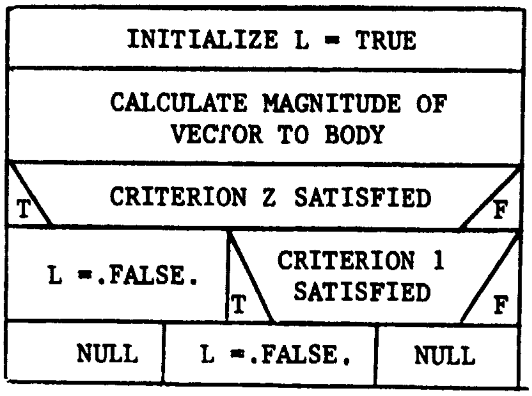


Figure A-9

```
0001          SUBROUTINE OCCULT(RSB,RSS,RB,L)
         C*************************************************************************
         C         THIS ROUTINE DETERMINES IF SATELLITE(S)IS BEING
         C         SHADED BY BODY B: RSS IS ECI VECTOR FROM SATELLITE TO
         C         TARGET, RSB IS THE ECI VECTOR TO THE BODY, AND RB IS THE
         C         OBSTRUCTED BODY RADIUS. L IS TRUE FOR SHADING
         C
         C         THE METHOD USES THE DOT PRODUCT OF RSS AND RSB AND
         C         COMPARES THE SINE OF THE INCLUDED ANGLE WITH THE
         C         SINE OF THE ANGLE OPPOSIT RB IN SUCH A WAY AS TO
         C         NOT NEED TRIG FUNCTIONS
         C
         C         CALLING ROUTINE    -    VISIBLE
         C
         C         CALLED ROUTINES    -    VMAG
         C                                 VDOT
         C
         C         CHECKED BY JACK MYERS 2JUNE1980
         C
         C*************************************************************************
0002          INCLUDE 'DEBUG.COM'
0003    *     COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                     IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                     IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
0004          DIMENSION RSB(3),RSS(3)
0005          REAL*8 RA,RB,RDR,RSB,RSS,VDOT,VMAG
0006          LOGICAL L
0007          L = .TRUE.
0008          RA = VMAG(RSB,3)
0009          RDR = VDOT(RSB,RSS,3)
0010          IF(RA**2-RDR**2/VMAG(RSS,3)**2 .GE. RB*RB)L=.FALSE.
         C*************************************************************************
         C         TO COVER POSSIBLE PROBLEMS ARISING FROM THE SQUARING
         C         PROCESS ABOVE
         C*************************************************************************
0011          IF(RDR .LE. 0.)L=.FALSE.
0012          IF (IDEBUG .GT. 3) WRITE(6,999) RSB,RSS,RB,L
0013     999  FORMAT(4X,' ON EXITING OCCULT ',7F13.2,L5)
0014          RETURN
0015          END
```

```
PROGRAM SECTIONS

    Name                              Bytes   Attributes

  0 $CODE                               199   PIC CON REL LCL   SHR   EXE    RD NOWRT LONG
  1 $PDATA                               35   PIC CON REL LCL   SHR NOEXE    RD NOWRT LONG
  2 $LOCAL                               84   PIC CON REL LCL NOSHR NOEXE    RD   WRT QUAD
  3 DEBUG                                 8   PIC OVR REL GBL   SHR NOEXE    RD   WRT LONG

ENTRY POINTS

    Address   Type  Name

  0-00000000        OCCULT

VARIABLES

    Address   Type  Name           Address   Type  Name           Address   Type  Name           Address   Type  Name

  3-00000004  I*4   IDEBUG       3-00000000  I*4   IENTER       AP-00000010@ L*4  L             2-00000000  R*8   RA
 AP-0000000C@ R*8   RB           2-00000008  R*8   RDR

ARRAYS

    Address   Type  Name                Bytes  Dimensions

 AP-00000004@ R*8   RSB                   24  (3)
 AP-00000008@ R*8   RSS                   24  (3)

LABELS

    Address    Label

  1-00000000   99'

FUNCTIONS AND SUBROUTINES REFERENCED

VDOT          VMAG

Total Space Allocated = 326 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

A-62

COMPILATION STATISTICS

```
  Run Time:              0.87 seconds
  Elapsed Time:          9.62 seconds
  Page Faults:           351
  Dynamic Memory:        160 pages
```

A-63

2.2.1.1.1 Compute vector to star (BVECT)

The function of this module is to compute a unit vector in inertial space pointing along the boresight of the Kth star tracker where K is an input. The process, illustrated by the VCLR in Figure A-10, begins by establishing a unit vector in star tracker coordinates defined by

$$\underline{U} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Note that the z axis of the star trackers point along the boresight. This vector is transformed into body coordinates using a two step process. First, the vector is transformed from star tracker coordinates to nominal star tracker coordinates to account for known misalignments. The resulting vector is transformed from nominal star tracker coordinates to body coordinates to account for orientation of the sensor.

The transformation from body coodinates to inertial coordinates is computed by the subroutine AMAT. This transformation matrix is then used to transform the boresight vector into inertial coordinates.

Figure II-3 BVECT

Form unit vector in star tracker coordinates along the sensor boresight
Transform unit vector into nominal star tracker coordinates to account for known misalignment
AMAT Compute transformation from body to inertial coordinates
Transform unit vector in body coordinates into inertial coordinates
Return vector to star in inertial coordinates

Figure A-10

```
              SUBROUTINE BVECT(VECT,K)
        C************************************************************
        C
        C     THE FUNCTION OF SUBROUTINE BVECT IS TO ESTABLISH A UNIT
        C     VECTOR ALONG THE INSTRUMENT BORESIGHT FOR THE KTH STAR
        C     TRACKER. THE VECTOR WILL BE KNOWN IN INERTIAL SPACE.
        C
        C     INPUT PARAMETERS
        C             K        = AN INTEGER DEFINING THE STAR TRACKER
        C                          TO BE USED
        C
        C     OUTPUT PARAMETERS
        C             VECT     = A UNIT VECTOR KNOWN IN INERTIAL SPACE
        C                          DEFINING THE BORESIGHT OF THE
        C                          KTH STAR TRACKER
        C
        C     WRITTEN BY JACK MYERS - 2JULY1980
        C                          EXT 4443
        C
        C************************************************************
              INCLUDE 'ENVIR.COM'
      •     COMMON /ENVIP/ STATE(10),PROFILE(10,4),INIT
      •     REAL*8 STATE,PROFILE
      • C
      • C             REAL WORLD STATE PARAMETERS
      • C
      • C                  STATE       STATE VALUES: X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
      • C                  PROFILE     ATTITUDE PROFILE-TIME (SEC) VS
      • C                              INERTIAL ANGULAR RATES (RAD/SEC)
      • C                  INIT        INTEGRATION INITIALIZATION KEY (-1)
      • C
              INCLUDE 'STARPAR.COM'
      •       COMMON /STARPAR/ BS(2,2),SS(2,2),TNS(3,3,2),TBNS(3,3,2),
      •               BSK(2,2),SSK(2,2),TNSK(3,3,2)
      •       REAL*8 BS,SS,TNS,TBNS,BSK,SSK,TNSK
      • C
      • C     STAR TRACKER PARAMETERS
      • C             IN EACH CASE THE LAST SUBSCRIPT REFERS TO THE
      • C                     TRACKER USED
      • C             BS       = BIAS - ACTUAL (RAD)
      • C             SS       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
      • C             TNS      = MISALIGNMENT ARRAY - TRANSFORMATION FROM
      • C                          STAR TRACKER TO NOMINAL
      • C             TBNS     = ORIENTATION ARRAY - TRANSFORMATION FROM
      • C                          NOMINAL TO BODY
      • C             BSK      = BIAS - KNOWLEDGE (RAD)
      • C             SSK      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
      • C             TNSK     = MISALIGNMENT KNOWLEDGE ARRAY - TRANSFORMATION
      • C                          FROM STAR TRACKER TO NOMINAL
      • C
              EQUIVALENCE (Q,STATE(7))
              DIMENSION UNIT(3),UNIT2(3),A1(3,3),B1(3,3),VECT(3)
              REAL*8 VECT,UNIT,UNIT2,A1,B1,Q
        C************************************************************
        C     GENERATE A UNIT VECTOR IN TRACKER COORDINATES
        C************************************************************
              UNIT(1)=0.
```

```
              UNIT(2)=0.
              UNIT(3)=1.
      C**************************************************************
      C       TRANSFORM TO INERTIAL SPACE
      C**************************************************************
              CALL MATAB(TNSK(1,1,K),UNIT,UNIT2,3,3,1)
              CALL MATAB(TBNS(1,1,K),UNIT2,UNIT,3,3,1)
              CALL AMAT(Q,A1)
              DO 10 I=1,3
              DO 10 J=1,3
      10      B1(I,J)=A1(J,I)
              CALL MATAB(B1,UNIT,VECT,3,3,1)
              RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	143	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	8	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	316	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 STARPAR	560	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		BVECT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-000000C0	I*4	I	3-00000190	I*4	INIT	2-000000C4	I*4	J	AP-00000008@	I*4	K
3-00000030	R*8	Q									

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000030	R*8	A1	72	(3, 3)
2-00000078	R*8	B1	72	(3, 3)
4-00000000	R*8	BS	32	(2, 2)
4-C0000160	R*8	BSK	32	(2, 2)
3-00000050	R*8	PROFILE	320	(10, 4)
4-00000020	R*8	SS	32	(2, 2)
4-00000180	R*8	SSK	32	(2, 2)
3-00000000	R*8	STATE	80	(10)
4-000000D0	R*8	TBNS	144	(3, 3, 2)

```
  4-00000040   R*8   TNS          144  (3, 3, 2)
  4-000001A0   R*8   TNSK         144  (3, 3, 2)
  2-00000000   R*8   UNIT          24  (3)
  2-00000018   R*8   UNIT2         24  (3)
 AP-00000004@  R*8   VECT          24  (3)
```

LABELS

```
    Address   Label

      **      10
```

FUNCTIONS AND SUBROUTINES REFERENCED

```
AMAT            MATAB
```

Total Space Allocated = 1431 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

A-68

COMPILATION STATISTICS

```
  Run Time:            1.04 seconds
  Elapsed Time:        13.93 seconds
  Page Faults:         341
  Dynamic Memory:      160 pages
```

2.3 Generate Environment (GENENV)

The generate environment module (GENENV) is responsible for propagating the true vehicle navigation and attitude states and for generating the gyro output between measurements. The true vehicle state is used to generate the measurements and for determining the error in the estimated state. The gyro output, which is processed by the executive through a call to GYRO, is used to propagate the vehicle attitude between measurements.

GENENV is called by the executive module (GCP). During the first call, GENENV is called by the true navigation state from present time up to measurement time using the real-world integrator, RUNG. This process is performed in small increments determined by the default integration interval DEL.

During all subsequent calls, GENENV simply generates the gyro output over a specified sampling period, DTA, by calling GYROUT. Before exiting, GENENV updates the vehicle time word, TIME.

A VCLR of GENENV is contained in Figure A-11.

GENERATE ENVIRONMENT GENENV

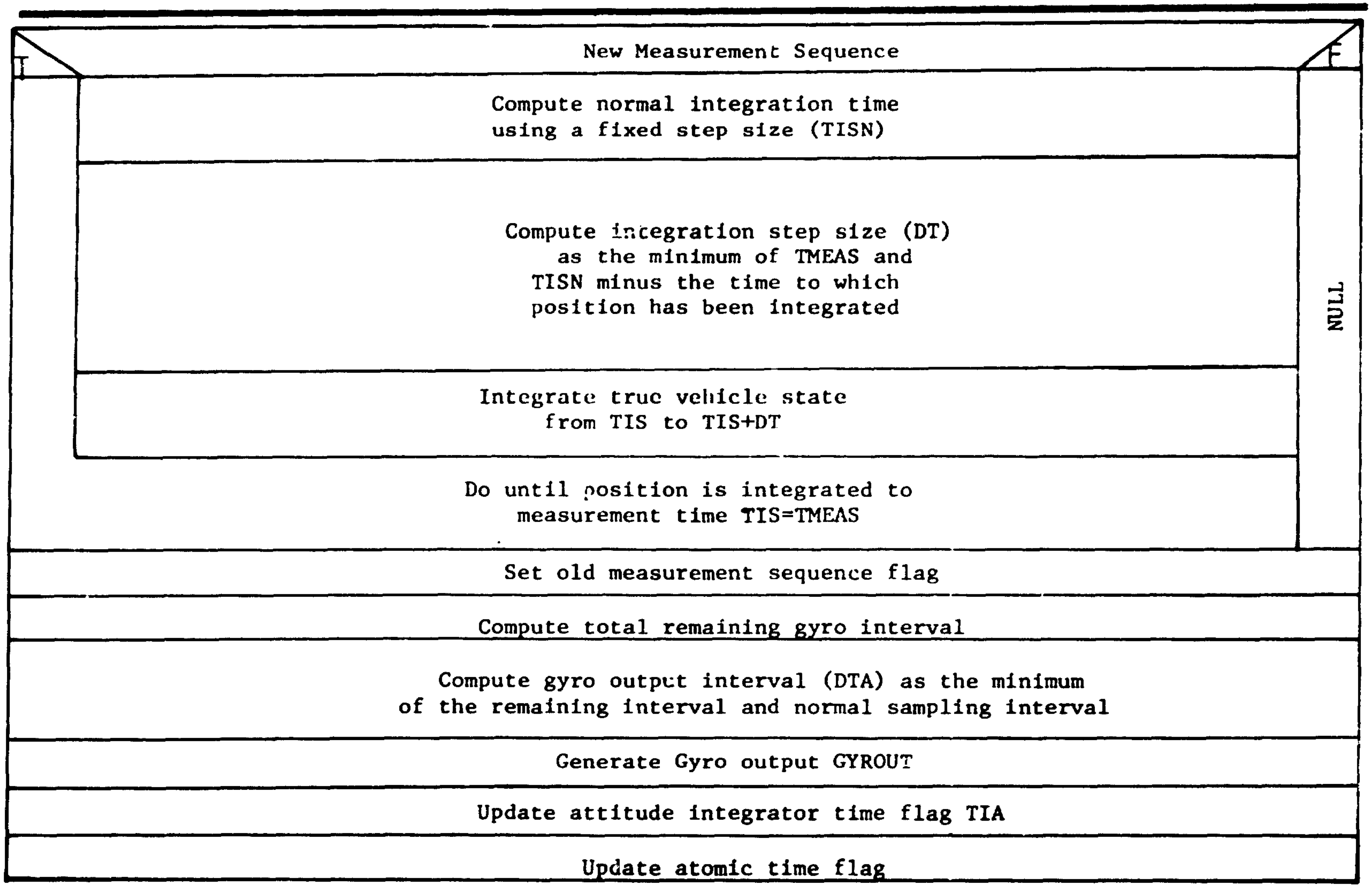


Figure A-11

```
      SUBROUTINE GENENV(FLAG)
      LOGICAL FLAG
      REAL*8 DGYRO,DT,DSTAT
      INCLUDE 'DEBUG.COM'
*           COMMON /DEBUG/ IENTER,IDEBUG
* C
* C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
* C
* C                      IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
* C                      IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
* C
              INCLUDE 'TIME.COM'
* C
*           COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
*          .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
*           REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
*          .         TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
* C
* C               THESE ARE THE TIME REFERENCE FRAMES
* C
* C                      TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
* C                      TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
* C                      TSTOP      RUN TERMINATION TIME (SEC)
* C                      TIA        ATTITUDE INTEGRATION TIME (SEC)
* C                      DEL          "         "         STEP SIZE (SEC)
* C                      TIN        POSITION INTEGRATION TIME (SEC)
* C                      DTN          "         "         STEP SIZE (SEC)
* C                      DATEO      DATE OF FLIGHT EPOCH (JD)
* C                      DATER      DATE OF 1950 EPOCH (JD)
* C                      TZERO      START TIME IN SECS. SINCE DATEO
* C                      TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
* C                      TIS        REAL WORLD REFERENCE TIME (SEC)
* C                      TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
* C                      DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                                 TOO LARGE AT MEASUREMENT TIME
* C                      TPRINT     TIME FOR PRINT (SEC)
* C                      DTPRINT    INCREMENT ON TPRINT (SEC)
* C
              INCLUDE 'TARGETS.COM'
*           COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
*           LOGICAL JFLAG
*           REAL*8 PI,TPI
* C
* C               MEASUREMENT SPECIFICATIONS
* C
* C                      MTYPE      MEASUREMENT TYPE
* C                      JFLAG      SET FOR STAR OBSTRUCTION
* C                      MCODE       "    "  MEASUREMENT PROCESSING
* C                      PI         PI
* C                      TPI        2*PI
* C
              INCLUDE 'ENVIR.COM'
*           COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
*           REAL*8 STATE,PROFILE
* C
* C               REAL WORLD STATE PARAMETERS
* C
```

```
  * C                     STATE     STATE VALUES: X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
  * C                     PROFILE   ATTITUDE PROFILE-TIME (SEC) VS
  * C                               INERTIAL ANGULAR RATES (RAD/SEC)
  * C                     INIT      INTEGRATION INITIALIZATION KEY (-1)
  * C
            INCLUDE 'CONTRL.COM'
  *       COMMON /CONTRL/ MOP,TINT
  *       REAL*8 TINT
  * C
  * C              PROGRAM CONTROL DESCRIPTORS FOR MULTIPLE RUNS
  * C
  * C                    MOP        MODE OF OPERATION
  * C                                 1 = PREFLIGHT SIMULATION
  * C                                 2 = POSTFLIGHT SIMULATION
  * C                                 3 = MONTE CARLO SIMULATION
  * C              TINT       NUMBER OF SECONDS OF FULL OPERATION PER CYCLE
  * C
            DIMENSION DSTATE(10)
            IF(FLAG) GO TO 10
    C*****************************************************************************
    C       THIS IS A NEW MEASUREMENT
    C
    C       INTEGRATE TRU  STATE TO MEASUREMENT TIME
    C*****************************************************************************
    5       CALL TREG(TIS,TISN)             ! TISN=TIS+DTU
            DT = MIN(TISN,TMEAS) -TIS
            IF(DT.GT..01) CALL RUNG(INIT,STATE,DSTATE,TIS,DT)
            IF(TIS + .01 .LT. TMEAS) GO TO 5
            FLAG = .TRUE.
    C*****************************************************************************
    C       GENERATE GYRO OUTPUT
    C*****************************************************************************
    10      DGYRO=TMEAS - TIA               ! COMPUTE INTERVAL REMAINING
            DTA = MIN(DEL,DGYRO)            ! COMPUTE OUTPUT INCREMENT
            CALL GYROUT                     ! GENERATE GYRO OUTPUT
            TIA=TIA + DTA                   ! UPDATE GYRO TIME WORD
            TIME = TIA
            RETURN
            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	139	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	132	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 TARGETS	36	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6 ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7 CONTRL	12	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GENENV

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
4-00000038	R*8	DATEO	4-00000070	R*8	DATER	4-00000020	R*8	DEL	2-00000050	R*8	DGYRO
2-00000058	R*8	DT	4-00000068	R*8	DTA	4-00000030	R*8	DTN	4-00000080	R*8	DTPRINT
AP-00000004●	L*4	FLAG	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	6-00000190	I*4	INIT
5-00000004	I*4	IS	5-0000000C	L*4	JFLAG	5-00000010	I*4	MCODE	7-00000000	I*4	MOP
5-00000000	I*4	MTYPE	5-00000008	I*4	NS	5-00000014	R*8	PI	4-00000018	R*8	TIA
4-00000000	R*8	TIME	4-00000028	R*8	TIN	7-00000004	R*8	TINT	4-00000058	R*8	TIS
4-00000060	R*8	TISN	4-00000048	R*8	TMEAS	4-00000008	R*8	TNEXT	5-0000001C	R*8	TPI
4-00000078	R*8	TPRINT	4-00000050	R*8	TRACK	4-00000010	R*8	TSTOP	4-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*8	DSTATE	80	(10)
6-00000050	R*8	PROFILE	320	(10, 4)
6-00000000	R*8	STATE	80	(10)

LABELS

Address	Label	Address	Label
0-0000000D	5	0-0000005D	10

FUNCTIONS AND SUBROUTINES REFERENCED

GYROUT RUNG TREG

Total Space Allocated = 867 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          1.10 seconds
  Elapsed Time:      19.39 seconds
  Page Faults:       340
  Dynamic Memory:    160 pages
```

2.3.1 INTEGRATE ACTUAL NAVIGATION STATE (RUNG)

The position state is advanced in time by numerical integration of the equations of motion. The second order equations of motion are composed of the external forces acting on the spacecraft. The external forces consist of geopotential, lunar gravitation and solar gravitation and radiation pressure.

A study showed that the Runge Kutta Gill (RKG) 4th order numerical integration method is optimal for this application. it is self-starting, handles variable step sizes, and is sufficiently accurate.

The Runge Kutta Gill (RKG) method for numerically integrating differential equations is described here.

The change in the value of the function during the computing interval is calculated by

$$\Delta y = \frac{1}{6}\{\kappa_1 + 2(1-k)\kappa_2 + 2(1+k)\kappa_3 + \kappa_4\}$$

where

$$\kappa_1 = h\cdot f(t_n, y_n) \qquad k = \sqrt{2}/2$$

$$\kappa_2 = h\cdot f(t_n + \tfrac{1}{2}h,\ y_n + \tfrac{1}{2}\kappa_1)$$

$$\kappa_3 = h\cdot f(t_n + \tfrac{1}{2}h,\ y_n + (-\tfrac{1}{2} + k)\kappa_1 + (1-k)\kappa_2)$$

$$\kappa_4 = h\cdot f(t_n + h,\ y_n - k\kappa_2 + (1+k)\kappa_3)$$

h = computing interval (seconds)

t_n = time of beginning of computing interval (seconds)

y_n = value of function at beginning of computing interval

The derivative function f shall be evaluated four times to calculate the change in the function being integrated during the computing interval.

At first appearance, the software algorithm has no relation to the mathematical description of the integrator. The following is intended to show that the two are indeed identical.

Proceeding through the outer do loop for K = 1 to 4 we find the following results:

K = 1 $\qquad f = f(t_n, y_n)$

$$t_1 = \tfrac{1}{2}f$$

$$y_1 = y_0 + \tfrac{1}{2}dtf$$

$$q_1 = f$$

K = 2 $\qquad f' = f(t_n + \frac{dt}{2},\ y_0 + \frac{dt}{2}f)$

$$t_2 = (1 - \frac{\sqrt{2}}{2})(f' - f)$$

$$y_2 = y_0 + \frac{dt}{2}f + dt(1 - \frac{\sqrt{2}}{2})(f' - f)$$

$$= y_0 + dt(-\tfrac{1}{2} + \frac{\sqrt{2}}{2})f + dt(1 - \frac{\sqrt{2}}{2})f'$$

$$q_2 = f + 3\ (1 - \frac{\sqrt{2}}{2})(f' - f)\ - (1 - \frac{\sqrt{2}}{2})f'$$

$$= f - 3(1 - \frac{\sqrt{2}}{2})f + 2(1 - \frac{\sqrt{2}}{2})f'$$

$$K = 3 \qquad f'' = f\left(t_n + \frac{dt}{2}, y + dt\left(-\frac{2}{2} + \frac{\sqrt{2}}{2}\right)f + dt\left(1 - \frac{\sqrt{2}}{2}\right)f'\right.$$

$$t_3 = \left(1 + \frac{\sqrt{2}}{2}\right)\left(f'' - f + 3\left(1 - \frac{\sqrt{2}}{2}\right)f - 2\left(1 - \frac{\sqrt{2}}{2}\right)f'\right)$$

$$y_3 = y_0 + dt\left(-\frac{1}{2} + \frac{\sqrt{2}}{2}\right)f + dt\left(1 - \frac{\sqrt{2}}{2}\right)f' + dt\left(1 + \frac{\sqrt{2}}{2}\right)\left(f'' - f + 3\left(1 - \frac{\sqrt{2}}{2}\right)f\right.$$

$$\left. -2\left(1 - \frac{\sqrt{2}}{2}\right)f'\right)$$

$$= y_0 - dt\frac{\sqrt{2}}{2}f' + dt\left(1 + \frac{\sqrt{2}}{2}\right)f''$$

$$q_3 = f - 3\left(1 - \frac{\sqrt{2}}{2}\right)f + 2\left(1 - \frac{\sqrt{2}}{2}\right)f' + 3\left(\left(1 + \frac{\sqrt{2}}{2}\right)\left\{f'' - f + 3\left(1 - \frac{\sqrt{2}}{2}\right)f - 2\left(1 - \frac{\sqrt{2}}{2}\right)f'\right\}\right.$$

$$\left. -\left(1 + \frac{\sqrt{2}}{2}\right)f''\right)$$

$$= -\frac{1}{2}f - \left(1 + \frac{\sqrt{2}}{2}\right)f' + 2\left(1 + \frac{\sqrt{2}}{2}\right)f''$$

$$K = 4 \qquad f''' = f\left(t_n + dt, y_0 - dt\frac{\sqrt{2}}{2}f' + dt\left(1 + \frac{\sqrt{2}}{2}\right)f''\right)$$

$$t_4 = \frac{1}{6}\left(f''' - 2 - \frac{1}{2}f - \left(1 + \frac{\sqrt{2}}{2}\right)f' + 2\left(1 + \frac{\sqrt{2}}{2}\right)f''\right)$$

$$= \frac{1}{6}\left(f''' + f + 2\left(1 + \frac{\sqrt{2}}{2}\right)f' - 4\left(1 + \frac{\sqrt{2}}{2}\right)f''\right)$$

$$y_4 = y_0 - dt\frac{\sqrt{2}}{2}f' + dt\left(1 + \frac{\sqrt{2}}{2}\right)f'' + \frac{1}{6}\left(dtf + 2dt\left(1 + \frac{\sqrt{2}}{2}\right)f' - 4dt\left(1 + \frac{\sqrt{2}}{2}\right)f'' + dtf'''\right.$$

$$= y_0 - \frac{1}{6}\left(dtf + 2dt\left(1 - \frac{\sqrt{2}}{2}\right)f' + 2dt\left(1 + \frac{\sqrt{2}}{2}\right)f'' + dtf'''\right)$$

Making the substitutions

$$K_1 = dtf$$
$$K_2 = dtf' \qquad k = \frac{\sqrt{2}}{2}$$
$$K_3 = dtf''$$
$$K_4 = dtf'''$$

Yields

$$y - y_0 = \Delta y = \frac{1}{6}\{K_1 + 2(1 - k)K_2 + 2(1 + k)K_3 + K_4\}$$

which is identical to the original algorithm.

RUNG VCLR

INITIAL ENTRY

T: NULL Q MATRIX

F: NULL

COMPUTE POSITION DYNAMICS

K = 1 OR 3

T: T = T + DT/2

F: NULL

TP = AA(K) * (ZD(I) - BB(K) * Q(I))

W = Z(I)

Z(I) = Z(I) + TP * DT

TP = (Z(I) - W)/DT

Q(I) = Q(I) + 3*T - CC(K) * ZD(I)

Where AA = .5, 1-$\sqrt{2}$/2, 1 + $\sqrt{2}$/2, .5/3 BB = 2,1,1,2
CC = .5, 1- $\sqrt{2}$/2, 1 + $\sqrt{2}$/2, .5

DO UNTIL I = 6

DO UNTIL K = 4

Figure A-12

```
0001          SUBROUTINE RUNG(INIT,Z,ZD,TI,DT)
        C*******************************************************************
        C        THIS ROUTINE INTEGRATES THE SIX REAL WORLD NAVIGATION
        C        STATES FROM TIME TI TO TI + DTU USING A RUNGE KUTTA GILL
        C        FORMULATION. STATE PARTIALS,ZD , ARE GENERATED BY DNAV
        C        AND INTEGRATED TO FORM Z.
        C*******************************************************************
0002          INCLUDE 'DEBUG.COM'
0003    *     COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                    I NTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                    IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
0004          DIMENSION Z(6),ZD(6)
0005          DIMENSION AA(4),BB(4),CC(4),Q(7)
0006          REAL*8 DT,T,TI,AA,BB,CC,Q,Z,ZD
0007          DATA AA /.5,.292893,1.707107,.1666667/
0008          DATA BB /2.,1.,1.,2./
0009          DATA CC /.5,.292893,1.707107,.5/
        C*******************************************************************
        C        INITIALIZE Q ARRAY
        C*******************************************************************
0010          DO 10 I=1,6
0011    10    Q(I) = 0.
        C*******************************************************************
        C        COMPUTE FOUR TERMS OF RUNGE KUTTA INTEGRATOR
        C*******************************************************************
0012          DO 30 K=1,4
0013             CALL DNAV(TI,Z,ZD,IDUMMY) ! COMPUTE POS. DYNAMIC PARTIALS
0014             IF(K.EQ.1.OR.K.EQ.3) TI = TI + DT/2.
        C*******************************************************************
        C        INTEGRATE SIX NAVIGATION STATES
        C*******************************************************************
0015             DO  30 I=1,6
0016                T = AA(K)*(ZD(I)-BB(K)*Q(I))
0017                Z(I) = Z(I)+T*DT
0018                Q(I) = Q(I)+3.*T-CC(K)*ZD(I)
0019    30     CONTINUE
0020          RETURN
0021          END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	169	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	232	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		RUNG

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
AP-00000014@	R*8	DT	2-000000A0	I*4	I	3-00000004	I*4	IDEBUG	2-000000A8	I*4	IDUMMY
3-00000000	I*4	IENTER	AP-00000004@	I*4	INIT	2-000000A4	I*4	K	2-00000098	R*8	T
AP-00000010@	R*8	TI									

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*8	AA	32	(4)
2-00000020	R*8	BB	32	(4)
2-00000040	R*8	CC	32	(4)
2-00000060	R*8	Q	56	(7)
AP-00000008@	R*8	Z	48	(6)
AP-0000000C@	R*8	ZD	48	(6)

LABELS

Address	Label	Address	Label
**	10	**	30

FUNCTIONS AND SUBROUTINES REFERENCED

DNAV

Total Space Allocated = 409 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           1.02 seconds
  Elapsed Time:       13.59 seconds
  Page Faults:        332
  Dynamic Memory:     160 pages
```

2.3.1.1 SPECIFY INTEGRATION STEP SIZE (TREG)

This module generates the endpoint of the integration period using the default interval DTV.

SPECIFY INTEGRATION STEP

COMPUTE MAXIMUM TIME FOR NEXT POSITION INTEGRATION TNEXT = TIME + DTV

Figure A-13

```
      SUBROUTINE TREG(TIME,TNEXT)
      INCLUDE 'DEBUG.COM'
•             COMMON /DEBUG/ IENTER,IDEBUG
• C
• C                 USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
• C
• C                        I NTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
• C                        IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
• C
              INCLUDE 'RVEC.COM'
•             COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
•                          ,RA,R2,R3,RSMA,RTG(3)
•             REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
• C
• C                 THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
• C
• C                        R          EARTH CENTER TO S/C -   ECI (KM)
• C                        RM           "      "      MOON -   ECI (KM)
• C                        RO           "      "      SUN  -   ECI (KM)
• C                        RSM        SPACECRAFT TO MOON -       ECI (KM)
• C                        RSO            "         SUN -          ECI (KM)
• C                        RSS        EARTH CENTER TO STAR -  ECI
• C                        RA         ABSOLUTE OF VECTOR R (KM)
• C                        R2         SQUARE OF RA (KM 2)
• C                        R3         CUBE OF RA (KM 3)
• C                        RSMA       ABSOLUTE OF RSM (KM)
• C
              INCLUDE 'CONST.COM'
•             COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
•             REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
• C
• C                 PROGRAM CONSTANTS
• C
• C                        ATM        S/C AREA TO MASS RATIO (METERS/KG)
• C                        RBM        OBSTRUCTION RADIUS OF THE MOON (KM)
• C                        RBE             "          "            EARTH (KM)
• C                        RBO             "          "            SUN (KM)
• C                        R 2        SQUARE OF THE EARTHS RADIUS (KM 2)
• C                        RM2           "         LUNAR RADIUS (KM 2)
• C                        UM         LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
• C                        US         SOLAR        "           "        "
• C                        UE         EARTH        "           "        "
• C                        J2,J3,J4   ZONAL GRAVITATIONAL HARMONIC TERMS
• C                        DTU        REGULARIZED TIME STEP SIZE (SEC)
• C                        PKI        SOLAR PRESSURE CONSTANT
• C
  C            COMPUTE REGULARIZED TIME INTERVAL
              REAL*8 TIME,TNEXT
              TNEXT = TIME + DTU
              RETURN
              END
```

```
TREG                                    6-Apr-1981 14:49:00     VAX-11 FORTRAN V2.0-2              Page   2
                                        11-Sep-1980 11:58:50    _DBA0:[D11R.GCP]TREG.FOR;3

PROGRAM SECTIONS

    Name                                Bytes   Attributes

  0 $CODE                                  13   PIC CON REL LCL   SHR   EXE   RD NOWRT LONG
  3 DEBUG                                   8   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  4 RVEC                                  224   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  5 CONST                                 112   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG

ENTRY POINTS

    Address     Type  Name

  0-00000000          TREG

VARIABLES

    Address     Type  Name          Address     Type  Name          Address     Type  Name          Address     Type  Name

  5-00000000    R*8   ATM         5-00000060    R*8   DTU         3-00000004    I*4   IDEBUG      3-00000000    I*4   IENTER
  5-00000048    R*8   J2          5-00000050    R*8   J3          5-00000058    R*8   J4          5-00000068    R*8   PK1
  4-00000080    R*8   R2          4-00000088    R*8   R3          4-000000A8    R*8   RA          5-00000010    R*8   RBE
  5-00000008    R*8   RBM         5-00000018    R*8   RBO         5-00000020    R*8   RE2         5-00000028    R*8   RM2
  4-000000C0    R*8   RSMA       AP-00000004@   R*8   TIME       AP-00000008@   R*8   TNEXT       5-00000040    R*8   UE
  5-00000030    R*8   UM          5-00000038    R*8   US

ARRAYS

    Address     Type  Name                  Bytes  Dimensions

  4-00000000    R*8   R                        24  (3)
  4-00000018    R*8   RM                       24  (3)
  4-00000030    R*8   RO                       24  (3)
  4-00000048    R*8   RSM                      24  (3)
  4-00000060    R*8   RSO                      24  (3)
  4-00000078    R*8   RSS                      24  (3)
  4-000000C8    R*8   RTG                      24  (3)
  4-00000090    R*8   SB                       24  (3)

Total Space Allocated = 357 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:            0.66 seconds
  Elapsed Time:        12.52 seconds
  Page Faults:         271
  Dynamic Memory:      160 pages
```

2.3.1.2 COMPUTE POSITION DYNAMICS (DNAV)

This module computes the forces acting on the spacecraft and generates the differential equation to be integrated by the state propagator.

The position of the spacecraft is calculated by solving three simultaneous, second order differential equations:

$$\ddot{x}_1 = -x_1 \cdot \frac{\mu}{R^3} + g_1(t,x) + a_1(t,x)$$

$$\ddot{x}_2 = -x_2 \cdot \frac{\mu}{R^3} + g_2(t,x) + a_2(t,x)$$

$$\ddot{x}_3 = -x_3 \cdot \frac{\mu}{R^3} + g_3(t,x) + a_3(t,x)$$

where

$x = |x_1, x_2, x_3|^T$

μ = Earth gravity constant ($3.985491204E + 05$ Km^3/sec^2)

$R = (x_1{}^2 + x_2{}^2 + x_3{}^2)^{1/2}$

x_1, x_2, x_3 = Coordinate of spacecraft

g_1, g_2, g_3 = Accelerations caused by zonal harmonia of earth gravity

a_1, a_2, a_3 = Solar radiation pressure perturbations, sun and moon gravity

The zonal harmonic accelerations are computed by:

$$g_1 = -\mu \frac{x_1}{R^3} F_1$$

$$g_2 = -\mu \frac{x_2}{R^3} F_1$$

$$g_3 = -\frac{\mu}{x^2} \left| \frac{x_3}{R} F_1 - F_2 \right|$$

where

$$F_1 = \left| \left(\frac{Re}{R}\right)^2 \cdot f_2 J_2 + \left(\frac{Re}{R}\right)^3 \cdot f_3 J_3 + \left(\frac{Re}{R}\right)^4 \cdot f_4 J_4 \right|$$

$$F_2 = \left| \left(\frac{Re}{R}\right)^2 \cdot f_1 J_2 + \left(\frac{Re}{R}\right)^3 \cdot f_2 J_3 + \left(\frac{Re}{R}\right)^4 \cdot f_3 J_4 \right|$$

and

$$f_1 = -3 \frac{x_3}{R}$$

$$f_2 = -7.5\left(\frac{x_3}{R}\right)^2 + 1.5$$

$$f_3 = -17.5\left(\frac{x_3}{R}\right)^3 + 7.5\left(\frac{x_3}{R}\right)$$

$$f_4 = -39.375\left(\frac{x_3}{R}\right)^4 + 26.25\left(\frac{x_3}{R}\right)^2 - 1.875$$

J_2 = 1082.7E -6 (Harmonic term in earth gravity model)

J_3 = -2.56E -6 (Harmonic term in earth gravity model)

J_4 = -1.58E -6 (Harmonic term in earth gravity model)

The process shown in Figure A-14 begins by computing the Julian day corresponding to current time. This parameter is required to compute the solar and lunar ephemerides. Given the locations of the sun, moon, and spacecraft, radius vectors are generated to each of these bodies. These radius vectors are then used to compute solar pressure, lunar and solar gravitational perturbation, and local gravitational effects. These forces are then used to generate the 2nd order differential equations of motion previously discussed.

COMPUTE POSITION DYNAMICS

COMPUTE JULIAN DAY
COMPUTE SOLAR/LUNAR EPHEMERIDES
COMPUTE RADIUS VECTOR FROM S/C TO EARTH
COMPUTE RADIUS VECTOR FROM S/C TO MOON
COMPUTE RADIUS VECTOR FROM S/C TO SUN
COMPUTE SOLAR RADIATION PRESSURE
COMPUTE SOLAR/LUNAR PERTURBATION
COMPUTE GEOPOTENTIAL - 4 ZONAL HARMONICS
COMPUTE SECOND-ORDER EQUATIONS OF MOTION

Figure A-14

```
      SUBROUTINE DNAV(T,X,XD,ICALL)
      DIMENSION X(6),XD(6),ASP(3),ASLG(3)
      REAL*8 X,XD,ASP,ASLG,T,F2,T2,DATE,F3,F4,F1,RER,RER2,T1,ZR,ZR2
      REAL*8 VMAG
      INCLUDE 'DEBUG.COM'
      COMMON /DEBUG/ IENTER,IDEBUG
C
C           USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
C
C                 IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
C                 IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
C
      INCLUDE 'TIME.COM'
C
      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
     .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
C
C           THESE ARE THE TIME REFERENCE FRAMES
C
C                 TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
C                 TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
C                 TSTOP     RUN TERMINATION TIME (SEC)
C                 TIA       ATTITUDE INTEGRATION TIME (SEC)
C                 DEL          "         "      STEP SIZE (SEC)
C                 TIN       POSITION INTEGRATION TIME (SEC)
C                 DTN          "         "      STEP SIZE (SEC)
C                 DATEO     DATE OF FLIGHT EPOCH (JD)
C                 DATER     DATE OF 1950 EPOCH (JD)
C                 TZERO     START TIME IN SECS. SINCE DATEO
C                 TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
C                 TIS       REAL WORLD REFERENCE TIME (SEC)
C                 TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
C                 DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
C                           TOO LARGE AT MEASUREMENT TIME
C                 TPRINT    TIME FOR PRINT (SEC)
C                 DTPRINT   INCREMENT ON TPRINT (SEC)
C
      INCLUDE 'CONST.COM'
      COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
      REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4 DTU,PK1
C
C           PROGRAM CONSTANTS
C
C                 ATM       S/C AREA TO MASS RATIO (METERS/KG)
C                 RBM       OBSTRUCTION RADIUS OF THE MOON (KM)
C                 RBE          "          "        EARTH (KM)
C                 RBO          "          "        SUN (KM)
C                 RE2       SQUARE OF THE EARTHS RADIUS (KM 2)
C                 RM2          "      LUNAR RADIUS (KM 2)
C                 UM        LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
C                 US        SOLAR      "          "      "
C                 UE        EARTH      "          "      "
C                 J2,J3,J4  ZONAL GRAVITATIONAL HARMONIC TERMS
C                 DTU       REGULARIZED TIME STEP SIZE (SEC)
C                 PK1       SOLAR PRESSURE CONSTANT
```

```
      C
              INCLUDE 'RVEC.CC''
              COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
      *                     ,RA,R2,R3,RSMA,RTG(3)
      *
      *       REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
      C
      C               THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
      C
      C                       R          EARTH CENTER TO S/C -   ECI (KM)
      C                       RM           "      "     MOON -   ECI (KM)
      C                       RO           "      "     SUN -    ECI (KM)
      C                       RS         SPACECRAFT TO MOON -      ECI (KM)
      C                       R             "           SUN -        ECI (KM)
      C                       R          EARTH CENTER TO STAR -  ECI
      C                       R          ABSOLUTE OF VECTOR R (KM)
      C                       R2         SQUARE OF RA (KM 2)
      C                       R3         CUBE OF RA (KM 3)
      C                       RSMA       ABSOLUTE OF RSM (KM)
      C
      C ------ COMPUTE JULIAN DAY ----------
            DATE = DATE0+(T-TZERO)/86400.              ! COMPUTE JULIAN DATE
      C ------ COMPUTE SOLAR AND LUNAR EPHEMERIS ---------
            CALL EPHEM(DATE,RM,RO)
            DO 10 I=1,3
      C*************************************************************************
      C           ECI COORDINATES OF S/C (R), S/C-MOON (RSM), S/C-SUN (RSO)
      C*************************************************************************
            R(I) = -X(I)
            RSO(I) = RO(I)+R(I)
      10    RSM(I) = RM(I)+R(I)
            RA = VMAG(R,3)
            R2 = RA*RA
            R3 = RA*R2
            RSMA = VMAG(RSM,3)
      C*************************************************************************
      C           SOLAR PRESSURE COMPUTATIONS
      C*************************************************************************
            CALL SPRESS(ASP)
      C ------ FIRST TIME WRITE SOLAR PRESSURE AT TIME T ----------
            IF (IDEBUG .GT. 3) WRITE(6,998) TIN,ASP
      998   FORMAT(/4X,' ACC. DUE TO SOLAR PRESSURE (KM/SEC2) AT TIME ',F7.1,
           + 3E22.14)
      C*************************************************************************
      C           SOLAR LUNAR GRAVITATIONAL PERTURBATIONS
      C*************************************************************************
            CALL GPERT(X,ASLG)
      C ------ FIRST TIME WRITE GRAVITATIONAL PERTURBATIONS ----------
            IF (IDEBUG .GT. 3) WRITE(6,996) TIN,ASLG
      996   FORMAT(4X,' ACC. DUE TO MOON AND SUN AT TIME ',F7.1,12X,3E22.14)
      C*************************************************************************
      C           GEOPOTENTIAL - 4 ZONAL HARMONICS
      C*************************************************************************
            ZR = X(3)/RA
            ZR2 = ZR*ZR
            F1 = -3.*ZR
            F2 = 7.5*ZR2+1.5
            F3 = ZR*(7.5-17.5*ZR2)
```

```
      F4 = ZR2*(26.25-39.375*ZR2)-1.875
      RER = RBE/RA
      RER2 = RER*RER
      T1 = RER2*(F2*J2+RER*(F3*J3+RER*(F4*J4)))
      T2 = RER2*(F1*J2+RER*(F2*J3+RER*(F3*J4)))
C************************************************************************
C     COMPUTE STATE PARTIALS
C************************************************************************
      XD(1) = X(4)                                        ! X VELOCITY
      XD(2) = X(5)                                        ! Y VELOCITY
      XD(3) = X(6)                                        ! Z VELOCITY
      XD(4) = -UE*X(1)/R3*(T1+1.)+ASP(1)+ASLG(1)          ! X ACCELERATION

      XD(5) = -UE*X(2)/R3*(T1+1.)+ASP(2)+ASLG(2)          ! Y ACCELERATION
      XD(6) = -UE/R2*(ZR*(T1+1.)-T2)+ASP(3)+ASLG(3)       ! Z ACCELERATION
      IF (IDEBUG .GT. 3) WRITE(6,995) TIN,XD(4),XD(5),XD(6)
995   FORMAT(4X,' TOTAL ACCELERATION AT TIME ',F7.1,18X,3E22.14)
      IF (IDEBUG.GT.1) WRITE(6,994) (X(I),I=1,3), R
994   FORMAT(/,'      AT END OF DNAV, POSITION VECTOR',3(1X,E20.10),/,
     C '      AT END OF DNAV, SS TO EARTH     ',3(1X,E20.10))
      RETURN
      END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	736	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	255	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	288	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 CONST	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6 RVEC	224	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		DNAV

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-00000000	R*8	ATM	2-00000040	R*8	DATE	4-00000038	R*8	DATE0	4-00000070	R*8	DATER
4-00000020	R*8	DEL	4-00000068	R*8	DTA	4-00000030	R*8	DTN	4-00000080	R*8	DTPRINT
5-00000060	R*8	DTU	2-00000058	R*8	F1	2-00000030	R*8	F2	2-00000048	R*8	F3
2-00000050	R*8	F4	2-00000088	I*4	I	AP-00000010@	I*4	ICALL	3-00000004	I*4	IDEBUG
3-00000000	I*4	IENTER	5-00000048	R*8	J2	5-00000050	R*8	J3	5-00000058	R*8	J4
5-00000068	R*8	PK1	6-000000B0	R*8	R2	6-000000B8	R*8	R3	6-000000A8	R*8	RA
5-00000010	R*8	RBE	5-00000008	R*8	RBM	5-00000018	R*8	RBO	5-00000020	R*8	RE2

2-00000060	R*8	RER	2-00000068	R*8	RER2	5-00000028	R*8	RM2	6-000000C0	R*8	RSMA
AP-00000004@	R*8	T	2-00000070	R*8	T1	2-00000038	R*8	T2	4-00000018	R*8	TIA
4-00000000	R*8	TIME	4-00000028	R*8	TIN	4-00000058	R*8	TIS	4-00000060	R*8	TISN
4-00000048	R*8	TMEAS	4-00000008	R*8	TNEXT	4-00000078	R*8	TPRINT	4-00000050	R*8	TRACK
4-00000010	R*8	TSTOP	4-00000040	R*8	TZERO	5-00000040	R*8	UE	5-00000030	R*8	UM
5-00000038	R*8	US	2-00000078	R*8	ZR	2-00000080	R*8	ZR2			

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000018	R*8	ASLG	24	(3)
2-00000000	R*8	ASP	24	(3)
6-00000000	R*8	R	24	(3)
6-00000018	R*8	RM	24	(3)
6-00000030	R*8	RO	24	(3)
6-00000048	R*8	RSM	24	(3)
6-00000060	R*8	RSO	24	(3)
6-00000078	R*8	RSS	24	(3)
6-000000C8	R*8	RTG	24	(3)
6-00000090	R*8	SB	24	(3)
AP-00000008@	R*8	X	48	(6)
AP-0000000C@	R*8	XD	48	(6)

A-92

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	1-00000098	994'	1-0000006D	995'	1-0000003C	996'	1-00000000	998'

FUNCTIONS AND SUBROUTINES REFERENCED

FPHEM GPERT SPRESS VMAG

Total Space Allocated = 1759 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

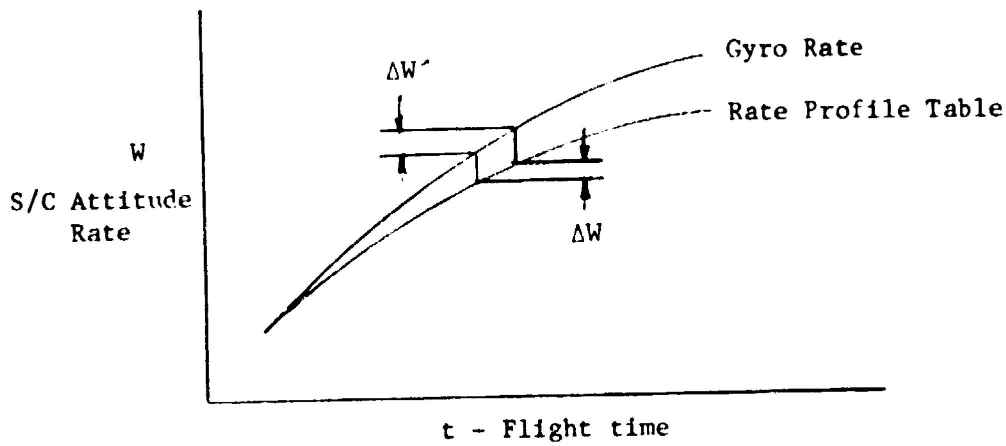
COMPILATION STATISTICS

Run Time: 2.63 seconds
Elapsed Time: 34.97 seconds
Page Faults: 407
Dynamic Memory: 160 pages

2.3.2 GENERATE GYRO OUTPUT (GYROUT)

The spacecraft attitude is determined from a tabular input of the S/C Euler angle rates as a function of time. This tabulation is called an attitude rate profile array. The gyro errors must be added to attitude rates calculated from the attitude rate profile to make these data realistic. Since the gyro drift effects are accumulative the gyro attitude diverges significantly from the profile attitude permitting only the determination of gyro updates to an accumulated gyro state.

The gyro drift error is accumulative and causes the output data to drift from the nominal attitude profile. To accommodate this drift effect a gyro attitude state is maintained and updated with the attitude profile. This effect is illustrated below.



Procedure:

1. Find the attitude rate from the profile table at time T_1.

2. Compute the change in rate since previous time.

3. Update the gyro rate and multiply the new rate by the scale factor bias error and then add a drift bias error

$$W = SF_{be} \cdot (W_0 + \Delta W) + Wd_{be}$$

4. Account for the random errors

$$W' = SF_{re} W + Wd_{re}$$

5. Compute the incremental gyro output data

$$\Delta W' = (W_1' - W_0')/DT$$

6. Add random error to the gyro misalignment term.

7. Transform to output frame.

GENERATE GYRO OUTPUT MODULE

Find actual attitude rate from profile rate table

LaGrange interpolation module (Module 3.3.1)

Find incremental change in attitude rate

Compute random and bias values of gyro drift and scale factor

Compute senced attitude rate

Find incremental change in actual attitude rate

Add increment to senced attitude rate

Adjust for scale factor and drift biases

Adjust for scale factor and drift variances

Multiply rate by incremental time (DTA)

Find rate gyro output

Find random value of nonorthogonal coefficient

Do until all three components of gyro output are computed

Compute nonorthogonal transformation matrix

Transform gyro output to update quaternions

Normalize quaternion coefficients

Figure A-15

```
        SUBROUTINE GYROUT
        DIMENSION TEMP(3,3)
        REAL*8 GD,DUMMY,OMEG,TEMP,GAUSS
        INCLUDE 'ARRAYS.COM'
* C
*       COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
*      .                ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
*       REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
* C
* C              THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
* C
* C                   T1 - T4      SINGLE DIMENSION ARRAYS
* C                   T11 - T77    DUAL DIMENSIONED ARRAYS
* C                   T11          DUAL ARRAY; OFF DIAGONAL SET TO ZERO
* C
        INCLUDE 'TIME.COM'
* C
*       COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
*      .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
*       REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
*      .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
* C
* C              THESE ARE THE TIME REFERENCE FRAMES
* C
* C                   TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
* C                   TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
* C                   TSTOP     RUN TERMINATION TIME (SEC)
* C                   TIA       ATTITUDE INTEGRATION TIME (SEC)
* C                   DEL         "           "       STEP SIZE (SEC)
* C                   TIN       POSITION INTEGRATION TIME (SEC)
* C                   DTN         "           "       STEP SIZE (SEC)
* C                   DATEO     DATE OF FLIGHT EPOCH (JD)
* C                   DATER     DATE OF 1950 EPOCH (JD)
* C                   TZERO     START TIME IN SECS. SINCE DATEO
* C                   TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
* C                   TIS       REAL WORLD REFERENCE TIME (SEC)
* C                   TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
* C                   DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                             TOO LARGE AT MEASUREMENT TIME
* C                   TPRINT    TIME FOR PRINT (SEC)
* C                   DTPRINT   INCREMENT ON TPRINT (SEC)
* C
        INCLUDE 'TMAT.COM'
*       COMMON /TMAT/ A(3,3),B(3,3),C(3,3),EM(4,3)
*       REAL*8 A,B,C,EM
* C
* C              TRANSFORMATION MATRICES
* C
* C                  A          INERTIAL TO BODY AXES
* C                  B          GYRO TO BODY AXES
* C                  C          GYRO NON-ORTHOGONAL TO GYRO AXES
* C                  EM         BODY TO QUATERNIAN AXES
* C
        INCLUDE 'NOISE.COM'
*       COMMON /NOISE/ BWD(3),SWD(3),BSF(3),SSF(3),BD(3),SD(3)
*      .               ,BDD(3),SDD(3),SRM,BRE,SRE
*       REAL*8 BWD,SWD,BSF,SSF,BD,SD,BDD,SDD,SRM,BRE,SRE
```

C-2

```
00800          * C
00800          * C                REAL WORLD  GYRO MEASUREMENT ERRORS
00800          * C
00800          * C       BWD  SWD          -          -        GYRO DRIFT (RAD / SEC)
00800          * C       BSF  SSF          -          -        GYRO SCALE FACTOR
00800          * C       BD   SD           -          -        GYRO NONORTHOGONALITY (RAD)
00800          * C       BDD  SDD          -          -        GYRO RELATIVE ORIENTATION (RAD)
00800          * C
00800    0016            INCLUDE 'ASTATE.COM'
00900          * C
00900    0017  *         COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
00900    0018  *         REAL*8 DE,E,WD,SF,D,DD
00900          * C
00900          * C                ATTITUDE STATE AND CONSIDERED PARAMETERS
00900          * C
00900          * C                     D.         DIFERENTIAL OF QUATERNIONS
00900          * C                     E          QUATERNIONS
00900          * C                     WD         GYRO DRIFT RATE (RAD/SEC)
00900          * C                     SF         GYRO SCALE FACTOR
00900          * C                     D          GYRO NON-ORTHOGANALITY (RAD)
00900          * C                     DD         GYRO RELATIVE ORIENTATION (RAD)
00900          * C
00900    0019            INCLUDE 'ENVIR.COM'
01000    0020  *         COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
01000    0021  *         REAL*8 STATE,PROFILE
01000          * C
01000          * C                REAL WORLD STATE PARAMETERS
01000          * C
01000          * C                     STATE      STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
01000          * C                     PROFILE    ATTITUDE PROFILE-TIME (SEC) VS
01000          * C                                INERTIAL ANGULAR RATES (RAD/SEC)
01000          * C                     INIT       INTEGRATION INITIALIZATION KEY (-1)
01000          * C
01000    0022            INCLUDE 'ROTAT.COM'
01100    0023  *         COMMON /ROTAT / DTHR(3),DTHEM(3),DTHE(3)
01100    0024  *         REAL*8 DTHR,DTH M,DTHE
01100          * C
01100          * C                GYRO ATTITUDE PARAMETERS
01100          * C
01100          * C                     DTHR       REAL WORLD GYRO DATA (RAD)
01100          * C                     DTHEM      FILTER WORLD GYRO DATA (RAD)
01100          * C                     DTHE       FILTER WORLD COMPENSATED GYRO DATA (RAD)
01100          * C
01100    0025            DIMENSION OMEG(3),DUMMY(3)
01200            C***************************************************************************
01300            C         INTERPOLATE TO FIND INTERTIAL RATES BETWEEN TIMES IN TABLE
01400            C***************************************************************************
01500    0026            DO 200 I = 1,10
01600    0027                 IF (PROFIL.(I,1).LT..001) GO TO 200
01700    0028                 IF (TIME.GT.PROFILE(I,1)) GO TO 200
01800    0029                 DO 100 J = 2,4
01900    0030                      OMEG(J-1) = PROFILE(I,J)
02000    0031                      IF (PROFILE(I,1)-PROFILE(I-1,1) .GT. 1.E-12)
02100                 *             OMEG(J-1) = ((TIME-PROFILE(I-1,1))/(PROFILE(I,1)-
02200                 *             PROFILE(I-1,1)))*(PROFILE(I,J)-PROFILE(I-1,J))+
02300                 *             PROFILE(I-1,J)
02400    0032       100            CONTINUE
```

```
               GO TO 205
  200      CONTINUE
 205   IF (IENTER.EQ.2)  WRITE(6,125)
  125 FORMAT(' ENTERING GYROUT')
       IF (IENTER.EQ.2)  WRITE(6,150) TIME,OMEG
  150 FORMAT(' TIME =',F8.1,' OMEG =',3(1X,E16.9))
C*************************************************************************
C       COMPUTE TRANSFORMATION MATRIX FROM GYRO TO BODY COORDINATES
C*************************************************************************
       CALL BMAT(BOD,T11)
C*************************************************************************
C       COMPUTE TRANSFORMATION MATRIX ACCOUNTING FOR GYRO NONORTHOGONALITY
C*************************************************************************
       CALL CMAT(BSF,BD,T33)
C
       IF (IENTER.EQ.2)
     . WRITE(6,175) ((T11(I,J),J=1,3),I=1,3),((T33(I,J),J=1,3),I=1,3)
  175 FORMAT(' B IS',/,3(3(1X,E16.9)),/,' C IS',/,3(3(1X,E16.9),/))
C*************************************************************************
C       FIND (INVERSE OF C)*(INVERSE OF B)=INVERSE OF (B*C). STORE IN TEMP
C*************************************************************************
       CALL MATAB(T11,T33,TEMP,3,3,3)
       CALL MINV3(TEMP,TEMP)
C*************************************************************************
C       FIND REAL WORLD CHANGE IN GYRO ANGLES
C*************************************************************************
       CALL MATAB(TEMP,OMEG,T1,3,3,1)
C*************************************************************************
C       CORRUPT REAL WORLD GYRO ANGLES WITH BIAS
C*************************************************************************
       DO 300 I = 1,3
  300      DTHR(I) = (BWD(I) + T1(I))*DTA
C
       IF (IENTER.EQ.2)
     . WRITE(6,180) ((TEMP(I,J),J=1,3),I=1,3),DTHR,DTHEM
  180 FORMAT(' INVERSE OF B X C',/,3(3(1X,E16.9),/),
     . ' REAL WORLD CHANGES IN GYRO ANGLES',/,3(1X,E16.9),/,
     . ' FILTER CHANGES IN GYRO ANGLES    ',/,3(1X,E16.9))
C*************************************************************************
C       FIND FILTER WORLD CHANGE IN GYRO ANGLES BY ADDING RANDOM ERROR
C       DUE TO DRIFT
C*************************************************************************
       DO 400 I = 1,3
           GD = GAUSS(0.0,SWD(I))
  400      DTHEM(I) = DTHR(I) + GD*DTA
C*************************************************************************
C       UPDATE REAL WORLD QUATERNION
C       NOTE: COMPENSATE WITH SAME
C       NOISE TERMS TO GET REAL WORLD
C*************************************************************************
       IF (IENTER.EQ.2)  WRITE(6,500)
  500 FORMAT(/,' CALL KATT FOR REAL WORLD ')
       CALL KATT(DTHR,BWC,BOD,BSF,BD,DUMMY,STATE(7))
C
       RETURN
       END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	576	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	271	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	328	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 ARRAYS	1096	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 TMAT	312	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6 NOISE	216	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7 ASTATE	160	PIC OVR REL GBL SHR NOEXE RD WRT LONG
8 ENVIR	404	PIC OVR REL GBL SHR NOEXE RD WRT LONG
9 ROTAT	72	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GYROUT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
6-000000C8	R*8	BRE	4-00000038	R*8	DATE0	4-00000070	R*8	DATER	4-00000020	R*8	DEL
4-00000068	R*8	DTA	4-00000030	R*8	DTN	4-00000080	R*8	DTPRINT	2-00000078	R*8	GD
2-00000080	I*4	I	2-00000088	I*4	IENTER	8-00000190	I*4	INIT	2-00000084	I*4	J
6-000000D0	R*8	SRE	6-000000C0	R*8	SRM	4-00000018	R*8	TIA	4-00000000	R*8	TIME
4-00000028	R*8	TIN	4-00000058	R*8	TIS	4-00000060	R*8	TISN	4-00000048	R*8	TMEAS
4-00000008	R*8	TNEXT	4-00000078	R*8	TPRINT	4-00000050	R*8	TRACK	4-00000010	R*8	TSTOP
4-00000040	R*8	TZERO									

ARRAYS

Address	Type	Name	Bytes	Dimensions
5-00000000	R*8	A	72	(3, 3)
5-00000048	R*8	B	72	(3, 3)
6-00000060	R*8	BD	24	(3)
6-00000090	R*8	BDD	24	(3)
6-00000030	R*8	BSF	24	(3)
6-00000000	R*8	BWD	24	(3)
5-00000090	R*8	C	72	(3, 3)
7-00000070	R*8	D	24	(3)
7-00000088	R*8	DD	24	(3)
7-00000000	R*8	DE	32	(4)
9-00000030	R*8	DTHE	24	(3)
9-00000018	R*8	DTHEM	24	(3)
9-00000000	R*8	DTHR	24	(3)
2-00000048	R*8	DUMMY	24	(3)
7-00000020	R*8	E	32	(4)
5-000000D8	R*8	EM	96	(4, 3)
2-00000060	R*8	OMEG	24	(3)

```
8-00000050  R*8  PROFILE        320  (10, 4)
6-00000078  R*8  SD              24  (3)
6-000000A8  R*8  SDD             24  (3)
7-00000058  R*8  SF              24  (3)
6-00000048  R*8  SSF             24  (3)
8-00000000  R*8  STATE           80  (10)
6-00000018  R*8  SWD             24  (3)
3-00000000  R*8  T1              24  (3)
3-00000098  R*8  T11             72  (3, 3)
3-00000018  R*8  T2              24  (3)
3-00000030  R*8  T3              24  (3)
3-000000E0  R*8  T33             72  (3, 3)
3-00000048  R*8  T4              80  (10)
3-00000128  R*8  T44            128  (4, 4)
3-000003E8  R*8  T5              32  (4)
3-00000408  R*8  T6              32  (4)
3-000001A8  R*8  T66            288  (6, 6)
3-00000428  R*8  T7              32  (4)
3-000002C8  R*8  T77            288  (6, 3)
2-00000000  R*8  TEMP            72  (3, 3)
7-00000040  R*8  WD              24  (3)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	100	1-00000000	125'	1-00000013	150'	1-00000033	175'	1-00000062	180'	0-00000083	200
0-00000088	205	**	300	**	400	1-000000E5	500'				

FUNCTIONS AND SUBROUTINES REFERENCED

```
BMAT        CMAT        GAUSS       KATT        MATAB       MINV3
```

Total Space Allocated = 3571 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           2.92 seconds
  Elapsed Time:       40.70 seconds
  Page Faults:        408
  Dynamic Memory:     160 pages
```

2.3.3 Transformation from Euler Angles (BMAT)

Subroutine BMAT uses DD(3) an array of Euler angles to generate the direction cosine array B(3,3). The B matrix is an array used for transformation of data from the gyro coordinate set to the body coordinate set.

Processing Requirements

The processing generates a standard Euler angle transformation with a rotational sequence starting from the gyro coordinates first about the z gyro axes, then the y axis, followed by the x axis. All rotations are considered positive.

Input Requirements

DD - Euler angle array

DD(1) = x-axis rotation (rad)
DD(2) = y-axis rotation (rad)
DD(3) = z-axis rotation (rad)

Output Requirements

B = Direction cosine transformation from gyro axes to body axes (unitless)

Subroutines and Function Called - SIN, COS (Math Package)

Calling Subroutines - GYROUT, KATT

"B" TRANSFORMATION MATRIX

FIND SINE AND COSINE OF EACH OF THE THREE COMPONENTS OF GYRO RELATIVE ORIENTATION
COMPUTE ELEMENTS OF B MATRIX USING THE ABOVE, WHERE B TRANSFORMS DATA FROM GYRO TO SEXTANT COORDINATES

Figure A-16

```
0001               SUBROUTINE BMAT(DD,B)
           C
           C            THIS ROUTINE COMPUTES THE B(3,3) MATRIX
           C             THE B MATRIX TRANSFORMS DATA FROM GYRO TO BODY COORDINATE
           C
0002               INCLUDE 'DEBUG.COM'
00100      0003  *        COMMON /DEBUG/ IENTER,IDEBUG
00200            * C
00300            * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400            * C
00500            * C                   IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600            * C                   IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
00700            * C
           0004           DIMENSION DD(3),B(3,3)
           0005           REAL*8 DD,B,CDX,CDY,CDZ,SDX,SDY,SDZ
           0006           CDZ = COS(DD(3))
           0007           SDZ = SIN(DD(3))
           0008           CDY = COS(DD(2))
           0009           SDY = SIN(DD(2))
           0010           CDX = COS(DD(1))
           0011           SDX = SIN(DD(1))
           0012           B(1,1) = CDZ*CDY
           0013           B(1,2) = SDZ*CDY
           0014           B(1,3) = -SDY
           0015           B(2,1) = CDZ*SDY*SDX-SDZ*CDX
           0016           B(2,2) = SDZ*SDY*SDX+CDZ*CDX
           0017           B(2,3) = CDY*SDX
           0018           B(3,1) = CDZ*SDY*CDX+SDZ*SDX
           0019           B(3,2) = SDZ*SDY*CDX-CDZ*SDX
           0020           B(3,3) = CDY*CDX
           0021           RETURN
           0022           END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	223	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	88	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		BMAT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	R*8	CDX	2-00000008	R*8	CDY	2-00000010	R*8	CDZ	3-00000004	I*4	IDEBUG
3-00000000	I*4	IENTER	2-00000018	R*8	SDX	2-00000020	R*8	SDY	2-00000028	R*8	SDZ

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	B	72	(3, 3)
AP-00000004@	R*8	DD	24	(3)

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DCOS MTH$DSIN

Total Space Allocated = 319 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 1.03 seconds
Elapsed Time: 14.26 seconds
Page Faults: 313
Dynamic Memory: 160 pages

2.3.4 Transformation from Quaternians (AMAT)

Subroutine AMAT completes the direction cosine matrix A(3,3) from the quaternian array E(4). The resulting A matrix may be used to transform data from the ECI to body coordinates.

Processing Requirements

The standard transformation equations from quaternian elements to A matrix elements are used. Refer to Motensen, RE "Strapdown Guidance Error Analysis" IEEE Transactions on Aerospace and Electronic Systems May 1974, pp 451-457

Input Requirements

E = The quaternian array describing the body attitude in inertial space (unitless)

Output Requirements

A = Direction cosine matrix for transformation of data from ECI to body coordinator. (unitless)

Subroutine and Functions Called - None

Calling Subroutines - BVECT, LAMKT, HLMT, HSTAR, EST

Mathematical Specification

The transformation from a quaternian array to a direction cosine array is given by

$$A = \begin{bmatrix} E_1^2 + E_2^2 - E_3^2 - E_4^2 & 2(E_2E_3 + E_1E_4) & 2(E_2E_4 - E_1E_3) \\ 2(E_2E_3 - E_1E_4) & E_1^2 - E_2^2 + E_3^2 - E_4^2 & 2(E_1E_2 + E_3E_4) \\ 2(E_2E_4 + E_1E_3) & 2(E_3E_4 - E_1E_2) & E_1^2 - E_2^2 - E_3^2 + E_4^2 \end{bmatrix}$$

Where the quaternian array is given by

$$E = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \end{bmatrix}$$

"A" TRANSFORMATION MATRIX

COMPUTE E COMPONENTS AS FUNCTIONS OF PRESENT QUATERNION

COMPUTE A MATRIX USING E COMPONENTS, WHERE A IS MATRIX REQUIRED TO TRANSFORM DATA FROM ECI TO SEXTANT COORDINATES

Figure A-17

```
      SUBROU[illegible]NE AMAT(E,A)
      DIMENSI[illegible] E(4),A(3,3)
      REAL*8 E,A,E00,E11,E22,[illegible]33,E[illegible],E02,E03,E12,E13,E23
C
C       THIS ROUTINE COMPUTES THE A(3,3) MATRIX
C       THE A MATRIX TRANSFORMS DATA FROM ECI TO SEXTANT COORNINATES
C
C
C         INPUT VARIABLES
C                 E = QUATERNIAN TRANSFORMATION FROM INERTIAL TO BODY
C
C         OUTPUT VARIABLES
C                 A = DIRECTION COSINE TRANSFORMATION FROM INERTIAL
C                     TO BODY
C
C         CODE CHECKED BY JACK MYERS 2JUNE1980
C
C
      INCLUDE 'DEBUG.COM'
*             COMMON /DEBUG/ IENTER,IDEBUG
* C
* C                    USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
* C
* C                          IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
* C                          IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
* C
              E00 = E(1)*E(1)
              E11 = E(2)*E(2)
              E22 = E(3)*E(3)
              E33 = E(4)*E(4)
              E01 = E(1)*E(2)
              E02 = E(1)*E(3)
              E03 = E(1)*E(4)
              E12 = E(2)*E(3)
              E13 = E(2)*E(4)
              E23 = E(3)*E(4)
              A(1,1) = E00+E11-E22-E33
              A(1,2) = 2.*(E12+E03)
              A(1,3) = 2.*(E13-E02)
              A(2,1) = 2.*(E12-E03)
              A(2,2) = E00-E11+E22-E33
              A(2,3) = 2.*(E01+E23)
              A(3,1) = 2.*(E13+E02)
              A(3,2) = 2.*(E23-E01)
              A(3,3) = E00-E11-E22+E33
              RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	216	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	120	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		AMAT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	R*8	E00	2-00000020	R*8	E01	2-00000028	R*8	E02	2-00000030	R*8	E03
2-00000008	R*8	E11	2-00000038	R*8	E12	2-00000040	R*8	E13	2-00000010	R*8	E22
2-00000048	R*8	E23	2-00000018	R*8	E33	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	A	72	(3, 3)
AP-00000004@	R*8	E	32	(4)

Total Space Allocated = 344 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DHAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 1.18 seconds
Elapsed Time: 11.63 seconds
Page Faults: 310
Dynamic Memory: 160 pages

2.4 PROCESS GYRO DATA (GYRO)

The general function of the Process Gyro Data Module is to process the angular changes sensed by the strapdown gyros at evenly spaced time intervals determined by one scan line of the science sensor.

This data is corrected for gyro considered parameters; drift, nonorthogonality, scale factor, and misalignment. It is then used to update the quaternion so that the attitude of the spacecraft can be known as accurately as possible at all times. Then, it is used to propagate the attitude state transition matrix. This matrix is used each time an attitude measurement is taken to propagate the covariance matrix.

The major input to the Process Gyro Data Module is the angular changes sensed by the gyros during each sample period. The length of this period defaults to .1 second, which is assumed to be the time it takes for one scan line of the MLA.

The input data consists of three variables which correspond to the angular changes sensed by the gyros about the three orthogonal gyro axes during the most recent period.

Other input data required by the Process Gyro Data Module are the initial value of the attitude quaternion and the time at which it is valid (the last attitude reference measurement time). The attitude quaternion is initially set with a default accuracy of .125 degrees about each axis.

The attitude quaternion, which is propagated by the Process Gyro Data Module, is later updated by the State Estimation Module when an attitude reference measurement is processed. It is also used by the State Estimation Module for calculating the predicted attitude reference measurement when updating the attitude quaternion. The three components of the attitude quaternion are in the closed interval [-1, +1].

The attitude state transition matrix which is progapated in the Process Gyro Data Module is later used in the State Estimation Module to determine an updated covariance matrix after an attitude reference measurement is processed. At this point, the attitude state transition matrix is reset to the identity.

The Process Gyro Data Module processes the gyro data to eliminate gyro systematic errors (process noise). The systematic errors include the constant drift rate, the scale factor, the non-orthogonality of the gyro input axes, and the misalignment of the ideal gyro input axes with respect to the spacecraft axis system. These compensated angular changes are calculated by the equation:

$$\begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \Delta\theta_3 \end{bmatrix} = B \cdot C \cdot \begin{bmatrix} \Delta\theta_{1m} + W_{Dx} \cdot \Delta t \\ \Delta\theta_{1m} + W_{Dy} \cdot \Delta t \\ \Delta\theta_{3m} + W_{Dz} \cdot \Delta t \end{bmatrix}$$

where

B = gyro misalignment transformation matrix

C = gyro scale factor and non-orthogonality transition matrix

$\Delta\theta_1$, $\Delta\theta_2$, $\Delta\theta_3$ = compensated gyro data (radians)

$\Delta\theta_{1m}$, $\Delta\theta_{2m}$, $\Delta\theta_{3m}$ = input data from gyros (radians)

W_{Dx}, W_{Dy}, W_{Dz} = gyro constant drift rate (rad/sec)

Δt = gyro measurement interval (sec)

The matrix C has the following components:

$$C = \begin{bmatrix} Sx & 0 & 0 \\ -Sx\cdot\delta_1 & Sy & 0 \\ -Sx\cdot\delta_2 & -Sy\cdot\delta_3 & Sz \end{bmatrix}$$

where

S_x, S_y, S_z = gyro scale factors (radians/count)

δ_1, δ_2, δ_3 = gyro misalignment angles (radians)

The matrix B has the following components:

$$\begin{bmatrix} C\Delta_3\cdot C\Delta_2 & S\Delta_3\cdot C\Delta_2 & -S\Delta_2 \\ S\Delta_1 . S\Delta_2\cdot S\Delta_3 - C\Delta_1\cdot S\Delta_3 & S\Delta_1\cdot S\Delta_2\cdot S\Delta_3 + C\Delta_1 . C\Delta_3 & C\Delta_2\cdot S\Delta_1 \\ C\Delta_1\cdot S\Delta_2\cdot C\Delta_3 + S\Delta_1\cdot S\Delta_3 & C\Delta_1\cdot S\Delta_2\cdot S\Delta_3 - S\Delta_1\cdot C\Delta^3 & C\Delta_2 - C\Delta_1 \end{bmatrix}$$

where the abbreviations C and S are used for cosine and sine, respectively, and Δ_3, Δ_2, Δ_1 are Euler angles representing rotations about the z axis, y axis, and x axis, in that order.

The compensated gyro data is then used to compute the differential of the quaternion which is used to propagate the quaternion. The quaternion is used to update the attitude transition matrix. The net result of this module being an update in the SS attitude estimate and a new attitude transition matrix.

GYRO PROCESSING MODULE (GYRO)

Compensate for Gyro drift (KATT)
Compensate for Gyro nonorthogonality (KATT)
Compensate for Gyro coordinate misalignment (KATT)
Update quaternion with angular changes (KATT)
Compute state partials, attitude parameter partials, and attitude state transition submatrices (PDATT)
Load new attitude transition matrix (PATT)

Figure A-18

```
      SUBROUTINE GYRO
      INCLUDE 'DEBUG.COM'
*           COMMON /DEBUG/ IENTER,IDEBUG
* C
* C                  USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
* C
* C                            IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
* C                            IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
* C
            INCLUDE 'ROTAT.COM'
*           COMMON /ROTAT / DTHR(3),DTHEM(3),DTHE(3)
*           REAL*8 DTHR,DTHEM,DTHE
* C
* C                  GYRO ATTITUDE PARAMETERS
* C
* C                            DTHR       REAL WORLD GYRO DATA (RAD)
* C                            DTHEM      FILTER WORLD GYRO DATA (RAD)
* C                            DTHE       FILTER WORLD COMPENSATED GYRO DATA (RAD)
* C
            INCLUDE 'ASTATE.COM'
* C
*           COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
*           REAL*8 DE,E,WD,SF,D,DD
* C
* C                  ATTITUDE STATE AND CONSIDERED PARAMETERS
* C
* C                            D          DIFERENTIAL OF QUATERNIONS
* C                            E          QUATERNIONS
* C                            WD         GYRO DRIFT RATE (RAD/SEC)
* C                            SF         GYRO SCALE FACTOR
* C                            D          GYRO NON-ORTHOGANALITY (RAD)
* C                            DD         GYRO RELATIVE ORIENTATION (RAD)
* C
            INCLUDE 'PHIA.COM'
*           COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
*        .            COVA(16,16),POA(16,16),QMAX
*           REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
* C
* C                  THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
* C
* C                            PA         ATTITUDE STATE TRANSITION MATRIX
* C                            TA            "       PARAMETER TRANSITION MATRIX
* C                            PDA           "       DERIVATIVE OF TRANSITION MATRICES
* C                            PHIA          "       AGGREGATE TRANSITION MATRIX
* C                            COVA          "       NEW COVARIANCE MATRIX
* C                            POA           "       PREVIOUS COVARIANCE MATRIX
* C                            QMAX          "       COVARIANCE NORM MAX
* C
            IF (IENTER.EQ.1) WRITE(6,999)
  999       FORMAT(' ENTERING GYRO ')
C***********************************************************************
C              UPDATE FILTER QUTERNION
C***********************************************************************
            IF (IENTER.EQ.2)  WRITE(6,100)
  100       FORMAT(/,' CALLING KATT FOR FILTER WORLD')
C***********************************************************************
C              KATT
```

```
      C       COMPENSATE FOR GYRO DRIFT
      C       COMPENSATE FOR NONORTHOGONALITY
      C       COMPENSATE FOR GYRO MISSALIGNMENT
      C       UPDATE QUATERNION
      C*****************************************************************
0017        CALL KATT(DTHEM,WD,DD,SF,D,DTHE,E)
      C*****************************************************************
      C       PDATT
      C       COMPUTE STATE PARTIALS
      C*****************************************************************
0018        CALL PDATT
      C*****************************************************************
      C       LOAD NEW STATE TRANSITION MATRIX
      C*****************************************************************
0019        CALL PATT(PHIA,PA,TA)
0020        RETURN
0021        END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	94	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	52	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	48	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	LONG
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 ROTAT	72	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 ASTATE	160	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6 PHIA	7176	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GYRO

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name
3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	6-00001C00	R*8	QMAX

ARRAYS

Address	Type	Name	Bytes	Dimensions
6-00000C00	R*8	COVA	2048	(16, 16)
5-00000070	R*8	D	24	(3)
5-00000088	R*8	DD	24	(3)
5-00000000	R*8	DE	32	(4)
4-00000030	R*8	DTHE	24	(3)

```
  4-00000018  R*8  DTHEM            24  (3)
  4-00000000  R*8  DTHR             24  (3)
  5-00000020  R*8  E                32  (4)
  6-00000000  R*8  PA              128  (4, 4)
  6-00000200  R*8  PDA             512  (4, 16)
  6-00000400  R*8  PHIA           2048  (16, 16)
  6-00001400  R*8  POA            2048  (16, 16)
  5-00000058  R*8  SF               24  (3)
  6-00000080  R*8  TA              384  (4, 12)
  5-00000040  R*8  WD               24  (3)
```

LABELS

Address	Label	Address	Label
1-00000012	100'	1-00000000	999'

FUNCTIONS AND SUBROUTINES REFERENCED

```
MA[illegible]        PATT        PDATT
```

Tot[illegible] [illegible] Allocated = 7610 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          1.04 seconds
Elapsed Time:      12.51 seconds
Page Faults:       326
Dynamic Memory:    160 pages
```

2.4.1 PROPAGATE ATTITUDE (KATT)

This module finds the differential of the quaternion for the last gyro period and uses it to update the quaternion (i.e., propagate the attitude estimate).

The differential of the quaternion is calculated by multiplying the compensated gyro data in sextant coordinates by the sextant to quaternion transition matrix. This transition matrix is:

$$EM = \frac{1}{2}\begin{bmatrix} -e_1 & -e_2 & -e_3 \\ e_0 & -e_3 & e_2 \\ e_3 & e_0 & -e_1 \\ -e_2 & e_1 & e_0 \end{bmatrix}$$

where e_i is the i^{th} element of the attitude quaternion.

This differential is then added to the quaternion to determine the new attitude.

ATTITUDE PROPAGATION (KATT)

COMPUTE COMPENSATION OF GYRO OUTPUT FOR RELATIVE ORIENTATION (BMAT)
COMPUTE COMPENSATION OF GYRO OUTPUT FOR NONORTHOGONALITY (CMAT)
COMPUTE TRANSFORMATOR FROM INDIVIDUAL GYRO TO S/C COORDINATES TEMP = BMAT * CMAT
COMPENSATE FOR GYRO DRIFT DTH - DRIFT
TRANSFORM GYRO OUTPUT INTO BODY COORDINATES DTHUP = TEMP * TI
COMPUTE CHANGE IN QUATERNION
UPDATE QUATERNION (QMULT)
UNITIZE QUATERNION (UNIT)

Figure A-19

```
0001            SUBROUTINE KATT(DTH,DRIFT,RELOR,SCALEF,NONOR,DTHUP,QUAT)
             C*********************************************************************
             C      THIS SUBROUTINE UPDATES THE REAL AND FILTER WORLD QUATERNIONS ABOUT
             C      THE BODY AXIS
             C*********************************************************************
0002            INCLUDE 'DEBUG.COM'
0003      *     COMMON /DEBUG/ IENTER,IDEBUG
          * C
          * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
          * C
          * C                   I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
          * C                   IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
          * C
0004            INCLUDE 'NOISE.COM'
0005      *     COMMON /NOISE/ BWD(3),SWD(3),BSF(3),SSF(3),BD(3),SD(3)
          *    .               ,BDD(3),SDD(3),SRM,BRE,SRE
0006      *     REAL*8 BWD,SWD,BSF,SSF,BD,SD,BDD,SDD,SRM,BRE,SRE
          * C
          * C             REAL WORLD  GYRO MEASUREMENT ERRORS
          * C
          * C     BWD  SWD        "        "       GYRO DRIFT (RAD / SEC)
          * C     BSF  SSF        "        "       GYRO SCALE FACTOR
          * C     BD   SD         "        "       GYRO NONORTHOGONALITY (RAD)
          * C     BDD  SDD        "        "       GYRO RELATIVE ORIENTATION (RAD)
          * C
0007            INCLUDE 'ARRAYS.COM'
          * C
0008      *     COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
          *    .                ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
0009      *     REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
          * C
          * C             THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
          * C
          * C                   T1 - T4      SINGLE DIMENSION ARRAYS
          * C                   T11 - T77    DUAL DIMENSIONED ARRAYS
          * C                   T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
          * C
0010              INCLUDE 'TIME.COM'
          * C
0011      *     COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
          *    .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0012      *     REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
          *    .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
          * C
          * C             THESE ARE THE TIME REFERENCE FRAMES
          * C
          * C                   TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
          * C                   TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
          * C                   TSTOP     RUN TERMINATION TIME (SEC)
          * C                   TIA       ATTITUDE INTEGRATION TIME (SEC)
          * C                   DEL         "          "        STEP SIZE (SEC)
          * C                   TIN       POSITION INTEGRATION TIME (SEC)
          * C                   DTN         "          "        STEP SIZE (SEC)
          * C                   DATEO     DATE OF FLIGHT EPOCH (JD)
          * C                   DATER     DATE OF 1950 EPOCH (JD)
          * C                   TZERO     START TIME IN SECS. SINCE DATEO
          * C                   TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
```

```
* C                      TIS       REAL WORLD REFERENCE TIME (SEC)
* C                      TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
* C                      DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                                TOO LARGE AT MEASUREMENT TIME
* C                      TPRINT    TIME FOR PRINT (SEC)
* C                      DTPRINT   INCREMENT ON TPRINT (SEC)
        DIMENSION DTH(3),DRIFT(3),RELOR(3),SCALEF(3),NONOR(3),DTHUP(3),
     +            QUAT(4),QTEMP(4),TEMP(3,3)
        REAL*8 DRIFT,DTH,DTHUP,QTEMP,QUAT,RELOR,SCALEF,TEMP,VDOT
  C
        IF (IENTER.GT.1) WRITE(6,999)
  999   FORMAT(' ENTERING KATT ')
  C************************************************************************
  C       COMPENSATE FOR RELATIVE ORIENTATION AND NONORTHOGONALITY
  C************************************************************************
        CALL BMAT(RELOR,T11)
        CALL CMAT(SCALEF,NONOR,T33)
        IF (IENTER.EQ.2)
       . WRITE(6,50) ((T11(I,J),J=1,3),I=1,3),((T33(I,J),J=1,3),I=1,3)
     50 FORMAT(' B IS',/,3(3(1X,E16.9),/),' C IS',/,3(3(1X,E16.9),/))
  C************************************************************************
  C       COMPUTE TRANSFORMATION FROM GYRO TO S/C COORDINATES
  C************************************************************************
        CALL MATAB(T11,T33,TEMP,3,3,3)
  C************************************************************************
  C       COMPENSATE FOR DRIFT BY SUBTRACTING ESTIMATED VALUE
  C************************************************************************
        DO 100 I = 1,3
  100      T1(I) = DTH(I) - DRIFT(I)*DTA
  C************************************************************************
  C       TRANSFORM GYRO RATES TO BODY RATES
  C************************************************************************
        CALL MATAB(TEMP,T1,DTHUP,3,3,1)
  C************************************************************************
  C       UPDATE QUATERNION
  C************************************************************************
        QTEMP(1) = 1.0 - VDOT(DTHUP,DTHUP,3)/8.
        DO 200 I = 2,4
  200      QTEMP(I) = .5*DTHUP(I-1)
        CALL QMULT(QUAT,QTEMP,QUAT)
        IF (IENTER.EQ.2)  WRITE(6,250) DTHUP,QUAT
  250   FORMAT(' DTHUP =',3(1X,E16.9),/,' QUAT =',4(1X,E16.9))
  C************************************************************************
  C       UNITIZE QUATERNION
  C************************************************************************
        CALL UNIT(QUAT,QUAT,4)
  C
        RETURN
        END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	475	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	118	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	384	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 NOISE	216	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 ARRAYS	1096	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6 TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		KATT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
4-000000C8	R*8	BRE	6-00000038	R*8	DATEO	6-00000070	R*8	DATER	6-00000020	R*8	DEL
6-00000068	R*8	DTA	6-00000030	R*8	DTN	6-00000080	R*8	DTPRINT	2-00000068	I*4	I
3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-0000006C	I*4	J	4-000000D0	R*8	SRE
4-000000C0	R*8	SRM	6-00000018	R*8	TIA	6-00000000	R*8	TIME	6-00000028	R*8	TIN
6-00000058	R*8	TIS	6-00000060	R*8	TISN	6-00000048	R*8	TMEAS	6-00000008	R*8	TNEXT
6-00000078	R*8	TPRINT	6-00000050	R*8	TRACK	6-00000010	R*8	TSTOP	6-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
4-00000060	R*8	BD	24	(3)
4-00000090	R*8	BDD	24	(3)
4-00000030	R*8	BSF	24	(3)
4-00000000	R*8	BWD	24	(3)
AP-00000008@	R*8	DRIFT	24	(3)
AP-00000004@	R*8	DTH	24	(3)
AP-00000018@	R*8	DTHUP	24	(3)
AP-00000014@	I*4	NONOR	12	(3)
2-00000000	R*8	QTEMP	32	(4)
AP-0000001C@	R*8	QUAT	32	(4)
AP-0000000C@	R*8	RELOR	24	(3)
AP-00000010@	R*8	SCALEF	24	(3)
4-00000078	R*8	SD	24	(3)
4-000000A8	R*8	SDD	24	(3)
4-00000048	R*8	SSF	24	(3)
4-00000018	R*8	SWD	24	(3)
5-00000000	R*8	T1	24	(3)
5-00000098	R*8	T11	72	(3, 3)
5-00000018	R*8	T2	24	(3)
5-00000030	R*8	T3	24	(3)
5-000000E0	R*8	T33	72	(3, 3)

```
5-00000048  R*8  T4       80  (10)
5-00000128  R*8  T44     128  (4, 4)
5-000003E8  R*8  T5       32  (4)
5-00000408  R*8  T6       32  (4)
5-000001A8  R*8  T66     288  (6, 6)
5-00000428  R*8  T7       32  (4)
5-000002C8  R*8  T77     288  (6, 6)
2-00000020  R*8  TEMP     72  (3, 3)
```

LABELS

```
Address     Label   Address  Label   Address  Label   Address     Label   Address     Label

1-00000012  50'     **       100     **       200     1-0000004'  250'    1-00000000  999'
```

FUNCTIONS AND SUBROUTINES REFERENCED

```
BMAT      CMAT      MATAB     QMULT     UNIT      VDCT
```

Total Space Allocated = 2433 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          1.83 seconds
Elapsed Time:      19.48 seconds
Page Faults:       390
Dynamic Memory:    160 pages
```

2.4.1.1 Matrix Multiply (MATAB)

Subroutine MATAB is a utility routine that performs a matrix multiply of the form

$$A(L,M) \times B(M,N) = C(L,N)$$

Algorithm

$$C(I,J) = \sum_{K=1}^{M} A(I,K)*B(K,J)$$

Input Variables

A = First matrix to be multiplied dimensioned LXM
B = Second matrix to be multiplier dimensioned MXN
L = Row dimension of A and C
M = Column dimension of A and row dimension of B
N = Column dimension of B and C

Output Variables

C = Resultant matrix dimensioned LXN

Subroutines and functions called - None

Calling Subroutines - Utility

MATAB VCLR

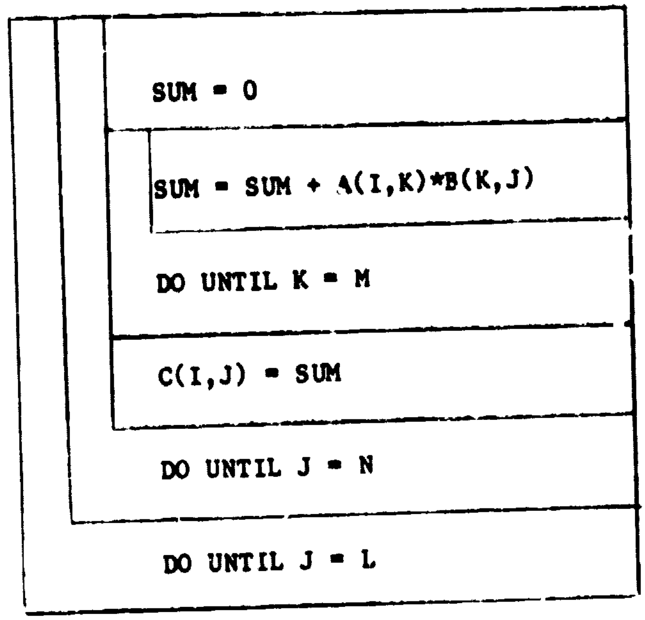


Figure A-20

```
0001            SUBROUTINE MATAB(A,B,C,L,M,N)
        C*****************************************************************
        C         MATRIX A(L,M) X B(M,N) = C(L,N)
        C*****************************************************************
0002            INCLUDE 'DEBUG.COM'
00100   0003  *          COMMON /DEBUG/ IENTER,IDEBUG
00200         * C
00300         * C                  USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400         * C
00500         * C                       IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600         * C                       IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
00700         * C
        0004             DIMENSION A(L,M),B(M,N),C(L,N)
        0005             REAL*8 SUM,A,B,C
        0006             DO 20 I=1,L
        0007             DO 20 J=1,N
        0008             SUM = 0.
        0009             DO 10 K=1,M
        0010      10     SUM = SUM+A(I,K)*B(K,J)
        0011      20     C(I,J) = SUM
        0012             RETURN
        0013             END
```


PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	216	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	184	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		MATAB

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000008	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-0000000C	I*4	J
2-00000010	I*4	K	AP-00000010@	I*4	L	AP-00000014@	I*4	M	AP-00000018@	I*4	N
2-00000000	R*8	SUM									

```
ARRAYS

    Address   Type  Name                 Bytes  Dimensions

 AP-00000004@ R*8   A                      **   (*, *)
 AP-00000008@ R*8   B                      **   (*, *)
 AP-0000000C@ R*8   C                      **   (*, *)


LABELS

    Address    Label        Address    Label

      **       10             **       20


Total Space Allocated = 408 Bytes


COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19


COMPILATION STATISTICS

  Run Time:             0.91 seconds
  Elapsed Time:         11.50 seconds
  Page Faults:          283
  Dynamic Memory:       160 pages
```

2.4.2 COMPUTE STATE PARTIALS (PDATT)

This module computes the state partials and the ϕ and θ submatrixes of the attitude transition matrix. These modules are loaded in the Load Transition Matrix module to form a new attitude transition matrix.

Two parts of the augmented state transition matrix Φ_A are updated each time the gyro data becomes available. The matrix Φ_A contains 1) a submatrix which represents the partial derivative of the quaternion with respect to the value of the quaternion at the time the last measurement is processed, and 2) a submatrix which represents the partial derivative of the quaternion with respect to gyro model parameters (drift, non-orthogonality, scale factor and misalignment). The mathematical equations are more easily read if the matrix Φ_A is partitioned

$$\Phi_A = \left[\begin{array}{c:c} \phi & \theta \\ \hdashline 0 & I \end{array}\right]$$

where

ϕ contains partial derivative of attitude quaternion with respect to the value of the quaternion at the last measurement time,

θ contains partial derivative of attitude quaternion with respect to gyro model parameters, and

I is an identity submatrix

The submatrix ϕ is updated each time the gyro data becomes available with the following equation

$$\phi(t_i + 1, t_0) = F' \phi(t_i, t_0)$$

where F is the state gradient matrix for attitude and

$$F' = \frac{1}{2} \begin{bmatrix} 0 & -\Delta\theta_1 & -\Delta\theta_2 & -\Delta\theta_3 \\ \Delta\theta_1 & 0 & \Delta\theta_3 & -\Delta\theta_2 \\ \Delta\theta_2 & -\Delta\theta_3 & 0 & \Delta\theta_1 \\ \Delta\theta_3 & +\Delta\theta_2 & -\Delta\theta_1 & 0 \end{bmatrix}$$

$\phi(t_i + 1, t_0)$ = state transition submatrix between times t_o and $t_i + 1$

θ_i = compensated angular changes sensed by the strapdown gyros.

The submatrix θ is updated each time the gyro data becomes available using the equation

$$\theta(t_i + 1, t_0) = F' \theta(t_i, t_0) + E$$

where E is the parameter gradient matrix for attitude

$$E = \frac{1}{2} \begin{bmatrix} -e_1 & -e_2 & -e_3 \\ e_0 & -e_3 & e_2 \\ e_3 & e_0 & -e_1 \\ -e_2 & e_1 & e_0 \end{bmatrix} \qquad \left[M_1 \;\vdots\; M_2 \;\vdots\; M_3 \;\vdots\; M_4 \right]$$

and the parts of the partitioned matrix are given by

$$M_1 = \begin{bmatrix} \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{bmatrix} \qquad \text{(drift)}$$

$$M_2 = \begin{bmatrix} 0 & 0 & 0 \\ -\Delta\theta_{1m} & 0 & 0 \\ 0 & -\Delta\theta_{1m} & -\Delta\theta_{2m} \end{bmatrix} \qquad \text{(non-orthogonality)}$$

$$M_3 = \begin{bmatrix} \Delta\theta_{1m} & 0 & 0 \\ 0 & \Delta\theta_{2m} & 0 \\ 0 & 0 & \Delta\theta_{3m} \end{bmatrix} \qquad \text{(scale factor)}$$

$$M_4 = \begin{bmatrix} 0 & -\Delta\theta_{m3} & \Delta\theta_{m2} \\ \Delta\theta_{m3} & 0 & -\Delta\theta_{m1} \\ -\Delta\theta_{m2} & \Delta\theta_{m1} & 0 \end{bmatrix} \qquad \text{(misalignment)}$$

where $\Delta\theta_{m1}$, $\Delta\theta_{m2}$, $\Delta\theta_{m3}$, = most recent gyro data.

The net results of this module are updated ϕ and θ submatrices.

2.4.3 LOAD ATTITUDE TRANSITION MATRIX (PATT)

This module uses the attitude state transition submatrix (ϕ) and the attitude parameter transition submatrix (θ) which were calculated in the Compute State Partials module as inputs to update the aggregated attitude transition matrix each time gyro data is received.

COMPUTE ATTITUDE STATE PARTIALS (PDATT)

COMPUTE F´ MATRIX (STATE GRADIENT MATRIX)

COMPUTE E (PARAMETER GRADIENT MATRIX)

COMPUTE SCALE FACTOR PORTION OF M

E´ = E´ * (I/I/M3/I)

COMPUTE NONORTHOGONALITY PORTION OF M

E´ = E´ * (I/M2/I/I)

COMPUTE DRIFT PORTION OF M

E´ = E´ * (M_1/I/I/I)

COMPUTE MISALIGNMENT PORTION OF M

E´ = E´ * (I/I/I/M4)

$$PDA = \left[\begin{array}{c|c} \phi(t_i + 1, t_0) & 0 \\ \hline 0 & I \end{array}\right] = \left[\begin{array}{c|c} F' * \phi(t_i, t_0) & 0 \\ \hline 0 & I \end{array}\right]$$

$$PDA = \left[\begin{array}{c|c} \phi & F'\theta' \\ \hline 0 & T \end{array}\right]$$

$$PDA = \left[\begin{array}{c|c} \phi & F'\theta' + E \\ \hline 0 & I \end{array}\right] = \left[\begin{array}{c|c} \phi & \theta \\ \hline 0 & I \end{array}\right]$$

ATTITUDE PARAMETER TRANSITION MATRIX TA = PDA

Figure A-21

```
0001            SUBROUTINE PDATT
          C
          C COMPUTES STATE PARTIALS AND FORMS PHI AND THETA SUBMATRIXES
          C OF THE ATTITUDE TRANSITION MATRIX WHERE
          C      1) PHI CONTAINS PARTIAL DERIVATIVES OF THE ATTITUDE QUATERNION
          C         WITH RESPECT TO LAST MEASUREMENT TIME, AND
          C      2) THETA CONTAINS PARTIAL DERIVATIVES OF ATTITUDE QUATERNION
          C         WITH RESPECT TO GYRO MODEL PARAMETERS.
          C
                INCLUDE 'DEBUG.COM'
0002            COMMON /DEBUG/ IENTER,IDEBUG
0003      *
          * C
          * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
          * C
          * C                  I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
          * C                  IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
          * C
                INCLUDE 'ASTATE.COM'
0004      * C
                COMMO. /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
0005      *     REAL*6  E,E,WD,SF,D,DD
0006      *
          * C
          * C              ATTITUDE STATE AND CONSIDERED PARAMETERS
          * C
          * C                  D          DIFERENTIAL OF QUATERNIONS
          * C                  E          QUATERNIONS
          * C                  WD         GYRO DRIFT RATE (RAD/SEC)
          * C                  SF         GYRO SCALE FACTOR
          * C                  D          GYRO NON-ORTHOGANALITY (RAD)
          * C                  DD         GYRO RELATIVE ORIENTATION (RAD)
          * C
          * C
                INCLUDE 'GFPART.COM'
0007            COMMON/GFPART/ FA(4,4),EA(4,12),FN(6,6)
0008      *     REAL*8 FA,EA,FN
0009      *
          * C
          * C              MEASUR.MENT AND STATE PARTIALS
          * C
          * C                  FA         ATTITUDE STATE PARTIALS
          * C                  EA             "    CONSIDERED PARAMETERS PARTIALS
          * C                  FN             "    STATE PARTIALS
          * C
          * C
                INCLUDE 'PHIA.COM'
00:0            COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
0011      *         COVA(16,16),POA(16,16),QMAX
          *     REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
0012      *
          * C
          * C              THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
          * C
          * C                  PA         ATTITUDE STATE TRANSITION MATRIX
          * C                  TA             "    PARAMETER TRANSITION MATRIX
          * C                  PDA            "    DERIVATIVE OF TRANSITION MATRICES
          * C                  PHIA           "    AGGREGATE TRANSITION MATRIX
          * C                  COVA           "    NEW COVARIANCE MATRIX
          * C                  POA            "    PREVIOUS COVARIANCE MATRIX
          * C                  QMAX           "    COVARIANCE NORM MAX
          * C
          * C
                INCLUDE 'ARRAYS.COM'
0013
```

```
        * C
0014    *        COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
        *       .         ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
0015    *        REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
        * C
        * C               THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
        * C
        * C                    T1 - T4      SINGLE DIMENSION ARRAYS
        * C                    T11 - T77    DUAL DIMENSIONED ARRAYS
        * C                    T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
        * C
0016             INCLUDE 'TMAT.COM'
0017    *        COMMON /TMAT/ A(3,3),B(3,3),C(3,3),EM(4,3)
0018    *        REAL*8 A,B,C,EM
        * C
        * C               TRANSFORMATION MATRICES
        * C
        * C                   A          INERTIAL TO BODY AXES
        * C                   B          GYRO TO BODY AXES
        * C                   C          GYRO NON-ORTHOGONAL TO GYRO AXES
        * C                   EM         BODY TO QUATERNIAN AXES
        * C
0019             INCLUDE 'ROTAT.COM'
0020    *        COMMON /ROTAT / DTHR(3),DTHEM(3),DTHE(3)
0021    *        REAL*8 DTHR,DTHEM,DTHE
        * C
        * C               GYRO ATTITUDE PARAMETERS
        * C
        * C                    DTHR       REAL WORLD GYRO DATA (RAD)
        * C                    DTHEM      FILTER WORLD GYRO DATA (RAD)
        * C                    DTHE       FILTER WORLD COMPENSATED GYRO DATA (RAD)
        * C
0022             INCLUDE 'TIME.COM'
        * C
0023    *        COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
        *       .         ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0024    *        REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
        *       .      TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
        * C
        * C               THESE ARE THE TIME REFERENCE FRAMES
        * C
        * C                    TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
        * C                    TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
        * C                    TSTOP      RUN TERMINATION TIME (SEC)
        * C                    TIA        ATTITUDE INTEGRATION TIME (SEC)
        * C                    DEL           "           "         STEP SIZE (SEC)
        * C                    TIN        POSITION INTEGRATION TIME (SEC)
        * C                    DTN           "           "         STEP SIZE (SEC)
        * C                    DATEO      DATE OF FLIGHT EPOCH (JD)
        * C                    DATER      DATE OF 1950 EPOCH (JD)
        * C                    TZERO      START TIME IN SECS. SINCE DATEO
        * C                    TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
        * C                    TIS        REAL WORLD REFERENCE TIME (SEC)
        * C                    TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
        * C                    DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
        * C                               TOO LARGE AT MEASUREMENT TIME
        * C                    TPRINT     TIME FOR PRINT (SEC)
```

```
* C                  DTPRINT   INCREMENT ON TPRINT (SEC)
* C
        REAL*8 DT1,DT2,DT3,E0,E1,E2,E3,TSLEW
        IF (IENTER.GT.1) WRITE(6,999)
  999   FORMAT(' ENTERING PDATT ')
        IF (IDEBUG.GT.3) WRITE(6,998) TSLEW
  998   FORMAT(4X,' IN PDATT, TSLEW = ',F20.6)
  C-----------------------------------------------------------------
  C COMPUTE F' STATE GRADIENT MATRIX
  C-----------------------------------------------------------------
        DT1 = .5*DTHE(1)
        DT2 = .5*DTHE(2)
        DT3 = .5*DTHE(3)
        FA(1,2) = -DT1
        FA(1,3) = -DT2
        FA(1,4) = -DT3
        FA(2,1) =  DT1
        FA(2,3) =  DT3
        FA(2,4) = -DT2
        FA(3,1) =  DT2
        FA(3,2) = -DT3
        FA(3,4) =  DT1
        FA(4,1) =  DT3
        FA(4,2) =  DT2
        FA(4,3) = -DT1
  C-----------------------------------------------------------------
  C COMPUTE BODY TO QUATERNION AXES TRANSFORMATION
  C-----------------------------------------------------------------
        E0 = .5*E(1)
        E1 = .5*E(2)
        E2 = .5*E(3)
        E3 = .5*E(4)
        EM(1,1) = -E1
        EM(1,2) = -E2
        EM(1,3) = -E3
        EM(2,1) =  E0
        EM(2,2) = -E3
        EM(2,3) =  E2
        EM(3,1) =  E3
        EM(3,2) =  E0
        EM(3,3) = -E1
        EM(4,1) = -E2
        EM(4,2) =  E1
        EM(4,3) =  E0
  C-----------------------------------------------------------------
  C COMPUTE SCALE FACTOR PORTION OF M, M3
  C-----------------------------------------------------------------
        DO 20 I=1,3
        DO 10 J=1,3
  10    T33(I,J) = 0.                          ! INITIALIZE T33 ARRAY
  20    T11(I,I) = DTHEM(I)
        CALL MATAB(EM,T11,EA(1,4),4,3,3)       ! E*[0/0/M3]
  C-----------------------------------------------------------------
  C COMPUTE NONORTHOGONALITY PORTION OF M, M2
  C-----------------------------------------------------------------
        T33(2,1) = -DTH_M(1)
        T33(3,2) = -DTH_M(1)
```

```
              T33(3,3) = -DTH_M(2)
              CALL MATAB(EM,T33,EA(1,7),4,3,3)          ! E*[0/M2/M3]
C-----------------------------------------------------------------------
C COMPUTE DRIFT PORTION OF M
C-----------------------------------------------------------------------
              DO 30 I=1,3
     30       T11(I,I) = - DTA
              CALL MATAB(EM,T11,EA,4,3,3)               ! E*[M1/M2/M3]
C-----------------------------------------------------------------------
C COMPUTE MISALIGNMENT PORTION OF M
C-----------------------------------------------------------------------
              T33(1,2) = -DTH_M(3)
              T33(1,3) =  DTH_M(2)
              T33(2,1) =  DTH_M(3)
              T33(2,3) = -DTH_M(1)
              T33(3,1) = -DTH_M(2)
              T33(3,2) =  DTH_M(1)
              T33(3,3) =  0.
              CALL MATAB(EM,T33,EA(1,10),4,3,3)         ! E[M1/M2/M3/M4]
              CALL MATAB(FA,PA,PDA,4,4,4)               ! DQ+DQ
              CALL MATAB(FA,TA,PDA(1,5),4,4,12)         ! F * THETA
              DO 40 I=1,4
              DO 40 J=5,16
              PDA(I,J) = EA(I,J-4)+PDA(I,J)             ! F * THETA + E
     40         CONTINUE
              DO 50 I=1,4
              DO 50 J=1,4
     50       PA(I,J) = PA(I,J)+PDA(I,J)                ! [PHI, THETA]
              DO 60 I=1,4
              DO 60 J=1,12
     60       TA(I,J) = TA(I,J)+PDA(I,J+4)
              RETURN
              END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	600	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	58	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	248	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3	DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4	ASTATE	160	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5	GFPART	800	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6	PHIA	7176	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7	ARRAYS	1096	PIC OVR REL GBL SHR NOEXE RD WRT LONG
8	TMAT	312	PIC OVR REL GBL SHR NOEXE RD WRT LONG
9	ROTAT	72	PIC OVR REL GBL SHR NOEXE RD WRT LONG
10	TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		PDATT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
10-00000038	R*8	DATEO	10-00000070	R*8	DATER	10-00000020	R*8	DEL	2-00000000	R*8	DT1
2-00000008	R*8	DT2	2-00000010	R*8	DT3	10-00000068	R*8	DTA	10-00000030	R*8	DTN
10-00000080	R*8	DTPRINT	2-00000018	R*8	E0	2-00000020	R*8	E1	2-00000028	R*8	E2
2-00000030	R*8	E3	2-00000040	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER
2-00000044	I*4	J	6-00001C00	R*8	QMAX	10-00000018	R*8	TIA	10-00000000	R*8	TIME
10-00000028	R*8	TIN	10-00000058	R*8	TIS	10-00000060	R*8	TISN	10-00000048	R*8	TMEAS
10-00000008	R*8	TNEXT	10-00000078	R*8	TPRINT	10-00000050	R*8	TRACK	2-00000038	R*8	TSLEW
10-00000010	R*8	TSTOP	10-00000040	R*8	TZERO						

ARRAYS

Address	Type	Name	Bytes	Dimensions
8-00000000	R*8	A	72	(3, 3)
8-00000048	R*8	B	72	(3, 3)
8-00000090	R*8	C	72	(3, 3)
6-00000C00	R*8	COVA	2048	(16, 16)
4-00000070	R*8	D	24	(3)
4-00000088	R*8	DD	24	(3)
4-00000000	R*8	DE	32	(4)
9-00000030	R*8	DTHE	24	(3)
9-00000018	R*8	DTHEM	24	(3)
9-00000000	R*8	DTHR	24	(3)
4-00000020	R*8	E	32	(4)
5-00000080	R*8	EA	384	(4, 12)
8-000000D8	R*8	EM	96	(4, 3)
5-00000000	R*8	FA	128	(4, 4)
5-00000200	R*8	FN	288	(6, 6)

```
6-00000000  R*8  PA      128  (4, 4)
6-00000200  R*8  PDA     512  (4, 16)
6-00000400  R*8  PHIA   2048  (16, 16)
6-00001400  R*8  POA    2048  (16, 16)
4-00000058  R*8  SF       24  (3)
7-00000000  R*8  T1       24  (3)
7-00000098  R*8  T11      72  (3, 3)
7-00000018  R*8  T2       24  (3)
7-00000030  R*8  T3       24  (3)
7-000000E0  R*8  T33      72  (3, 3)
7-00000048  R*8  T4       80  (10)
7-00000128  R*8  T44     128  (4, 4)
7-000003E8  R*8  T5       32  (4)
7-00000408  R*8  T6       32  (4)
7-000001A8  R*8  T66     288  (6, 6)
7-00000428  R*8  T7       32  (4)
7-000002C8  R*8  T77     288  (6, 6)
6-00000080  R*8  TA      384  (4, 12)
4-00000040  R*8  WD       24  (3)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	**	20	**	30	**	40	**	50	**	60
1-00000013	998'	1-00000000	999'								

FUNCTIONS AND SUBROUTINES REFERENCED

MATAB

Total Space Allocated = 10666 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          3.46 seconds
Elapsed Time:      62.49 seconds
Page Faults:       370
Dynamic Memory:    160 pages
```

2.4.3 LOAD ATTITUDE TRANSITION MATRIX (PATT)

This module uses the attitude state transition submatrix (ϕ) and the attitude parameter transition submatrix (θ) which were calculated in the Compute State Partials module as inputs to update the aggregated attitude transition matrix each time gyro data is received.

ALGORITHM AND PROCESS

The state transition matrix Φ_A is updated each time the attitude quaternion is updated, i.e., each time gyro data becomes available. The matrix Φ_A contains 1) a submatrix which represents the partial derivative of the quaternion with respect to the value of the quaternion at the time the last measurement is processed, and 2) a submatrix which represents the partial derivative of the quaternion with respect to gyro model parameters (drift, non-orthogonality, scale factor and misalignment). The Φ_A is partitioned as follows:

$$\Phi_A = \left[\begin{array}{c|c} \phi & \theta \\ \hline 0 & I \end{array}\right]$$

where ϕ contains partial derivative of attitude quaternion with respect to the value of the quaternion at the last measurement time

θ contains partial derivative of attitude quaternion with respect to gyro model parameters, and

I is an identity submatrix

ϕ and θ were calculated in the Computer State Partials module. They, along with I, are loaded together in this module to form the updated state transition matrix, Φ_A.

LOAD ATTITUDE TRANSITION MATRIX (PATT)

Initialize total transition matrix to 0.
Load attitude state transition matrix (Φ) into aggregated transition matrix.
Load attitude parameter transition matrix (Θ) into aggregated transition matrix.
Set remaining diagonal elements to 1.

Figure A-22

```
      SUBROUTINE PATT(PHIA,PA,TA)
      INCLUDE 'DEBUG.COM'
*           COMMON /DEBUG/ IENTER,IDEBUG
* C
* C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
* C
* C                     IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
* C                     IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
* C
            DIMENSION PHIA(16,16),PA(4,4),TA(4,12)
            REAL*8 PA,PHIA,TA
C*************************************************************************
C          LOAD THE TOTAL TRANSITION MATRIX
C*************************************************************************
            DO 10 I=1,16
            DO 10 J=1,16
   10       PHIA(I,J) = 0.
            DO 20 I=1,4
            DO 20 J=1,4
   20       PHIA(I,J) = PA(I,J)
            DO 30 I=1,4
            DO 30 J=5,16
   30       PHIA(I,J) = TA(I,J-4)
            DO 40 I=5,16
   40       PHIA(I,I) = 1.
            RETURN
            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	175	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	68	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		PATT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-00000004	I*4	J

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	PA	128	(4, 4)
AP-00000004@	R*8	PHIA	2048	(16, 16)
AP-0000000C@	R*8	TA	384	(4, 12)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
**	10	**	20	**	30	**	40

Total Space Allocated = 251 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 0.92 seconds
Elapsed Time: 11.55 seconds
Page Faults: 307
Dynamic Memory: 160 pages

2.5 INTEGRATE POSITION STATE (INTG)

The Navigation State Time Update module shall numerically integrate the differential equations of motion of the spacecraft and the linear differential equation for the state transition matrix. The fourth order Runge-Kutta-Gill method shall be used to perform the numerical integration.

The differential equations of motion require that the inertial accelerations of the spacecraft be calculated. These accelerations are due to

1. earth gravity

2. harmonics of earth gravity

3. solar gravity, solar pressure and lunar gravity

The ephemeris of the moon and sun must be determined to evaluate the lunar gravitational effects and solar perturbations.

The state transition matrix for the navigation state shall be integrated simultaneously with the equations of motion.

This is a procedural module in that it only orders events and does not perform calculations. The position state is integrated to the next specified time needed to restrict integration errors or the measurement time, whichever comes sooner. The maximum next integration step size is computed after each integration.

FILTER WORLD STATE INTEGRATOR (INTG)

Compute integration endpoint using a fixed step size (TNEXT)
Integration interval is the minimum of TNEXT and TMEAS minus current time
Integrate over interval (RKG)

Figure A-23

```
      SUBROUTINE INTG
      INCLUDE 'DEBUG.COM'
*            COMMON /DEBUG/ IENTER,IDEBUG
* C
* C                 USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
* C
* C                        I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
* C                        IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
* C
             INCLUDE 'TIME.COM'
* C
*            COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
*                 ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
*            REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
*                 TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
* C
* C                 THESE ARE THE TIME REFERENCE FRAMES
* C
* C                        TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
* C                        TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
* C                        TSTOP      RUN TERMINATION TIME (SEC)
* C                        TIA        ATTITUDE INTEGRATION TIME (SEC)
* C                        D L             "            "     STEP SIZE (SEC)
* C                        TIN        POSITION INTEGRATION TIME (SEC)
* C                        DTN             "            "     STEP SIZE (SEC)
* C                        DATEO      DATE OF FLIGHT EPOCH (JD)
* C                        DATER      DATE OF 1950 EPOCH (JD)
* C                        TZERO      START TIME IN SECS. SINCE DATEO
* C                        TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
* C                        TIS        REAL WORLD REFERENCE TIME (SEC)
* C                        TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
* C                        DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                                   TOO LARGE AT MEASUREMENT TIME
* C                        TPRINT     TIME FOR PRINT (SEC)
* C                        DTPRINT    INCREMENT ON TPRINT (SEC)
* C
             INCLUDE 'NSTATE.COM'
* C
*            COMMON /NSTATE/ XD(6),X(6),RADM,RADE
*            REAL*8 XD,X,RADM,RADE
* C
* C                  POSITION STATE AND CONSIDERED PARAMETERS
* C
* C                        XD         STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
* C                        X          STATE POSITION PARAMETERS (KM AND KM/SEC)
* C                        RADM       RADIUS OF THE MOON (KM)
* C                        RADE       EARTH DETECTABLE RADIUS (KM)
* C
* C
             DATA INIT/-1./
  C
             CALL TREG (TIN,TNEXT)
             DTN = MIN(TNEXT,TMEAS) - TIN
             IF(DTN.GT..01) CALL RKG(INIT,DTN)
             RETURN
             END
```

```
PROGRAM SECTIONS

    Name                                  Bytes   Attributes

  0 $CODE                                    67   PIC CON REL LCL   SHR   EXE   RD NOWRT LONG
  2 $LOCAL                                   28   PIC CON REL LCL NOSHR NOEXE   RD   WRT LONG
  3 DEBUG                                     8   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  4 TIME                                    136   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  5 NSTATE                                  112   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG

ENTRY POINTS

    Address   Type  Name

  0-00000000        INTG

VARIABLES

    Address   Type  Name             Address   Type  Name             Address   Type  Name             Address   Type  Name

  4-00000038  R*8   DATEO          4-00000070  R*8   DATER          4-00000020  R*8   DEL            4-00000068  R*8   DTA
  4-00000030  R*8   DTN            4-00000080  R*8   DTPRINT        3-00000004  I*4   IDEBUG         3-00000000  I*4   IENTER
  2-00000000  I*4   INIT           5-00000068  R*8   RADE           5-00000060  R*8   RADM           4-00000018  R*8   TIA
  4-00000000  R*8   TIME           4-00000028  R*8   TIN            4-00000058  R*8   TIS            4-00000060  R*8   TISN
  4-00000048  R*8   TMEAS          4-00000008  R*8   TNEXT          4-00000078  R*8   TPRINT         4-00000050  R*8   TRACK
  4-00000010  R*8   TSTOP          4-00000040  R*8   TZERO

ARRAYS

    Address   Type  Name                Bytes  Dimensions

  5-00000030  R*8   X                      48  (6)
  5-00000000  R*8   XD                     48  (6)

FUNCTIONS AND SUBROUTINES REFERENCED

RKG            TREG

Total Space Allocated = 351 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          0.74 seconds
Elapsed Time:      7.34 seconds
Page Faults:       329
Dynamic Memory:    160 pages
```

2.5.1 INTEGRATE POSITION PARAMETERS MODULE (RKG)

The Runge Kutta Gill (RKG) numerical integration method is used to integrate the position state and the position transition matrix simulaneously. It is a fourth order technique that adequately accounts for the nonlinear equations of motion for orbital flight and the higher order terms in the transition matrix.

The purpose of the GCPSIM navigation software is to incorporate filtered measurement data as artifical corrections to the state position vector. This requirement disqualifies multi-step techniques during the processing of measurement data since artificial corrections would violate the continuity of the inherent curve fits coefficients of these techniques.

A second consideration for measurement processng is provisions for variable time steps. Frequent navigation measurements will be needed during convergence followed by infrequent measurements needed to maintain a specified error tolerance. Changing the integration step size is simple and direct with single step methods, but difficult with multi-step methods. Variations in step size for multi-step methods is accomplished by interpolations and extrapolations for the Variable-Step/Variable-Order Methods and by intervening with a single step method to define the initial values at the new time increment for standard multi-step methods.

A study showed that the Runge Kutta Gill (RKG) 4th order numerical integration method is optimal for this application. It is self-starting, handles variable step sizes, and sufficiently accurate.

The Runge Kutta Gill (RKG) method for numerically integrating differential equations is described here.

The change in the value of the function during the computing interval is calculated by

where $$\Delta y = \frac{1}{6}\{K_1 + 2(1 - k)K_2 + 2(1 + k)K_3 + K_4\} \qquad k = \frac{\sqrt{2}}{2}$$

k_1 = $h \cdot f(t_n, y_n)$

k_2 = $h \cdot f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}K_1)$

k_3 = $h \cdot f(t_n + \frac{1}{2}h, y_n + (-\frac{1}{2} + k)K_1 + (1 - k)K_2)$

k_4 = $h.f(t_n + h, y_n - kK_2 + (1 + k)K_3)$

h = computing interval (seconds)

t_n = time of beginning of computing interval (seconds)

y_n = value of function at beginning of computing interval

The derivative function f shall be evaluated four times to calculate the change in the function being integrated during the computing interval.

INTEGRATE POSITION DYNAMICS MODULE (RKG)

T — Initial Entry — F

T	F
NULL Q Matrix	NULL

Compute Position Dynamics (5.2.1)

Compute Position Transition Matrix Derivatives (5.2.2)

T — K = 1 or 3 — F

T	F
T = T + DT/2	NULL

TP = AA(K) * (ZD(I) - BB(K)*Q(I))

W = Z(I)

Z(I) = Z(I) + TP * DT

TP = (Z(I) - W)/DT

Q(I) = Q(I) + 3 * TP - CC(K) * ZD(I)

Where AA = .5, $1-\sqrt{2}/2$, $1+\sqrt{2}/2$, .5/3 BB = 2, 1, 1, 2
CC = .5, $1-\sqrt{2}/2$, $1+\sqrt{2}/2$, .5

Do until I = 6

Do until K = 4

Figure A-24

```
      SUBROUTINE RKG(INIT,DT)
      INCLUDE 'DEBUG.COM'
      COMMON /DEBUG/ IENTER,IDEBUG
C
C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
C
C                    IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
C                    IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
C
      INCLUDE 'NSTATE.COM'
C
      COMMON /NSTATE/ XD(6),X(6),RADM,RADE
      REAL*8 XD,X,RADM,RADE
C
C              POSITION STATE AND CONSIDERED PARAMETERS
C
C                    XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
C                    X         STATE POSITION PARAMETERS (KM AND KM/SEC)
C                    RADM      RADIUS OF THE MOON (KM)
C                    RADE      EARTH DETECTABLE RADIUS (KM)
C
C
      INCLUDE 'PHIN.COM'
      COMMON /PHIN/ PN(6,6),PDN(6,6),PHIN(6,6),COVN(6,6),
     .         PON(6,6)
      REAL*8 PN,PDN,PHIN,COVN,PON
C
C              THESE ARE THE NAVIGATION TRANSITION AND COVARIANCE ARRAYS
C
C                    PN        POSITION STATE TRANSITION MATRIX
C                    PDN          "     DERIVATIVE OF TRANSITION MATRIX
C                    PHIN         "     AGGREGATE TRANSITION MATRIX
C                    COVN         "     NEW COVARIANCE MATRIX
C                    PON          "     PREVIOUS COVARIANCE MATRIX
C
      INCLUDE 'TIME.COM'
C
      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
     .         ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
C
C              THESE ARE THE TIME REFERENCE FRAMES
C
C                    TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
C                    TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
C                    TSTOP     RUN TERMINATION TIME (SEC)
C                    TIA       ATTITUDE INTEGRATION TIME (SEC)
C                    DEL          "          "        STEP SIZE (SEC)
C                    TIN       POSITION INTEGRATION TIME (SEC)
C                    DTN          "          "        STEP SIZE (SEC)
C                    DATEO     DATE OF FLIGHT EPOCH (JD)
C                    DATER     DATE OF 1950 EPOCH (JD)
C                    TZERO     START TIME IN SECS. SINCE DATEO
C                    TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
C                    TIS       REAL WORLD REFERENCE TIME (SEC)
C                    TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
```

```
* C                 DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                           TOO LARGE AT MEASUREMENT TIME
* C                 TPRINT    TIME FOR PRINT (SEC)
* C                 DTPRINT   INCREMENT ON TPRINT (SEC)
* C
        INCLUDE 'UPDT.COM'
* C
*       COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
*       REAL*8 QN,QA,Q,QDOT
* C
* C           STATE  STIMATION PARAMETERS
* C
* C                 QN        NAV. DYN. NOISE COVARIANCE DIAGONAL
* C                 QA        MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
* C                 Q         CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
* C                 QDOT      DIFFERENTIAL OF Q
* C
        DIMENSION AA(4),BB(4),CC(4),RNDOF1(6),RNDOF2(6,6),RNDOF3(6,6)
        REAL*8 DT,T,W,AA,BB,CC,RNDOF1,RNDOF2,RNDOF3
        DATA AA /.5,.292893,1.707107,.1666667/
        DATA BB /2.,1.,1.,2./
        DATA CC /.5,.292893,1.707107,.5/
        IF(INIT.EQ.0) GO TO 20
        DO 10 I=1,6
        RNDOF1(I) = 0.
        DO 10 J=1,6
        RNDOF2(I,J) = 0.
  10    RNDOF3(I,J) = 0.
        INIT = 0
  20    DO 50 K = 1,4
        CALL DNAV(TIN,X,XD,K)
        CALL PDNAV(PN,PDN)
        IF(K.EQ.1.OR.K.EQ.3) TIN = TIN + DT/2
        DO  30 I=1,6
        T = AA(K)*(XD(I)-BB(K)*RNDOF1(I))
        W = X(I)
        X(I) = X(I)+T*DT
C        T = (X(I)-W)/DT
  30    RNDOF1(I) = RNDOF1(I) + 3.*T - CC(K)*XD(I)
        DO 40 I=1,6
        DO 40 J=1,6
        T = AA(K)*(PDN(I,J)-BB(K)*RNDOF2(I,J))
        W = PN(I,J)
        PN(I,J) = PN(I,J)+T*DT
        T = (PN(I,J)-W)/DT
  40    RNDOF2(I,J) = RNDOF2(I,J) + 3.*T - CC(K)*PDN(I,J)
              DO 50 I = 1,6
                    DO 50 J = 1,6
                          T = AA(K)*(QDOT(I,J) - BB(K)*RNDOF3(I,J))
                          W = Q(I,J)
                          Q(I,J) = Q(I,J) + T*DT
                          T = (Q(I,J) - W)/DT
  50                      RNDOF3(I,J) = RNDOF3(I,J) + 3.*T - CC(K)*QDOT(I,J)
        RETURN
        END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	455	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	780	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 NSTATE	112	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 PHIN	1440	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7 UPDT	752	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		RKG

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
6-00000038	R*8	DATEO	6-00000070	R*8	DATER	6-00000020	R*8	DEL	AP-00000008@	R*8	DT
6-00000068	R*8	DTA	6-00000030	R*8	DTN	6-00000080	R*3	DTPRINT	2-000002E0	I*4	I
3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	AP-00000004@	I*4	INIT	2-000002E4	I*4	J
2-000002E8	I*4	K	4-00000068	R*8	RADE	4-00000060	R*8	RADM	2-00000200	R*8	T
6-00000018	R*8	TIA	6-00000000	R*8	TIME	6-00000028	R*8	TIN	6-00000058	R*8	TIS
6-00000060	R*8	TISN	6-00000048	R*8	TMEAS	6-00000008	R*8	TNEXT	6-00000078	R*8	TPRINT
6-00000050	R*8	TRACK	6-00000010	R*8	TSTOP	6-00000040	R*8	TZERO	2-00000208	R*8	W

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*8	AA	32	(4)
2-00000020	R*8	BB	32	(4)
2-00000040	R*8	CC	32	(4)
5-00000360	R*8	COVN	288	(6, 6)
5-00000120	R*8	PDN	288	(6, 6)
5-00000240	R*8	PHIN	288	(6, 6)
5-00000000	R*8	PN	288	(6, 6)
5-00000480	R*8	PON	288	(6, 6)
7-00000080	R*8	Q	288	(6, 6)
7-00000030	R*8	QA	128	(16)
7-00000100	R*8	QDOT	288	(6, 6)
7-00000000	R*8	QN	48	(6)
2-00000060	R*8	RNDOF1	48	(6)
2-00000090	R*8	RNDOF2	288	(6, 6)
2-00000180	R*8	RNDOF3	288	(6, 6)
4-00000030	R*8	X	48	(6)
4-00000000	R*8	XD	48	(6)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	0-0000003E	20	**	30	**	40	**	50

FUNCTIONS AND SUBROUTINES REFERENCED

DNAV PDNAV

Total Space Allocated = 3683 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,SMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19
```

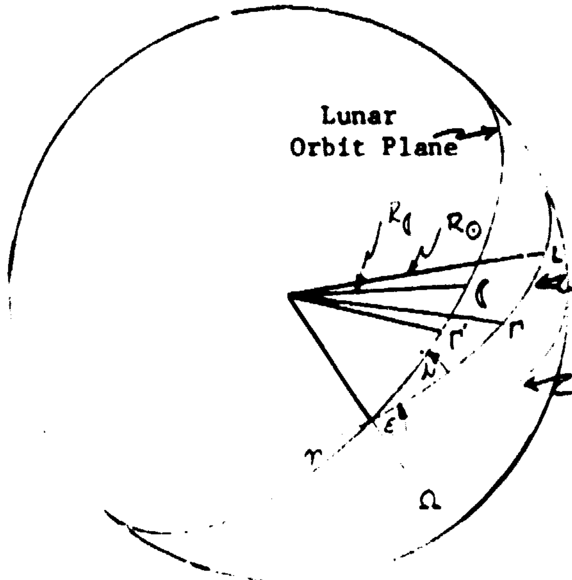
COMPILATION STATISTICS

```
Run Time:          2.40 seconds
Elapsed Time:      28.11 seconds
Page Faults:       381
Dynamic Memory:    160 pages
```

2.5.1.1 COMPUTE SOLAR/LUNAR EPHEMERIDES (EPHEM AND TRUEA)

The moon and sun positions are needed to compute gravitational perturbations, and star and lunar limb occultations. This module is used for both actual and filter calculations and for a given time in Julian days computes both positions in ECI coordinates to within 1 degree of arc.

The ephemerides of the sun and the moon are based upon the fundamental orbital elements for these bodies given in the Explanatory Supplement to the Nautical Almanac, pp. 98-107. These elements are expressed as second and third degree polynomials in time where the independent variable T is measured in Julian centuries of 36525 days. The epoch for both orbits is January 0.5 E.T. 1900.



Mean Orbital Elements of the Sun and Moon

Mean Orbital Elements of the Moon:

☾ = The mean longitude of the moon, measured in the ecliptic from the mean equinox of date to the mean ascending node of the lunar orbit, and then along the orbit;

Γ' = The mean longitude of the lunar perigee, measured in the ecliptic from the mean equinox of date to the mean ascending node of the lunar orbit, and then along the orbit;

Ω = The longitude of the mean ascending node of the lunar orbit on the ecliptic measured from the mean equinox of date;

e = The constant of eccentricity;

γ = Sin i/2, where i is the constant of inclination;

$\Pi_☾$ = The equatorial horizontal parallax at 60.2665 equatorial radii.

$☾ = 270^{0}26'02.''99 + 1336^{r}307^{0}52'59.''31T - 4.''08T^{2} + 0.''0068T^{3}$

$\Gamma' = 334^{0}19'46.''40 + 11^{r}109^{0}02'02.''52T - 37.''17T^{2} - 0.''045T^{3}$

$\Omega' = 259^{0}10'59.''79 - 5^{r}134^{0}08'31.''23T + 7.''48T^{2} + 0.''008T^{3}$

$e = 0.054900489$

$\gamma = 0.044886967 \quad (i = 5.^{0}145396366)$

$\Pi_☾ = 57'02.''70 \quad (\Pi = 0.0165932496 \text{ radians})$

<u>Mean Orbital Elements of the Sun:</u>

L = The geometric mean longitude of the sun measured in the ecliptic from the mean equinox of date

Γ = The mean longitude of perigee measured in the ecliptic frame the mean equinox of date

e = The mean eccentricity

ε = The mean obliquity of the ecliptic

Π_Θ = Horizontal parallax

$L = 279^{0}41'48.''04 + 129602768.''13T + 1.''089T^{2}$

$\Gamma = 281^{0}13'15.''00 + 6189.''03T + 1.''63T^{2} + 0.''0126T^{2}$

$e = 0.01675104 - 0.00004180T - 0.000000126T^{2}$

$\varepsilon = 23^{0}27'08.''26 - 46.''845T^{2} - 0.''0059T^{2} + 0.''00181T^{3}$

$\Pi_\Theta = 8.''80 \quad (\Pi_\Theta = 4.26624E-5 \text{ radians})$

The position vectors of the sun and moon in the ECI system are computed through the following chain of orthogonal transformations:

Moon position vector (ECI)

$$\left[\bar{R}_M\right] = \left[\varepsilon_{-X}\right]\left[\Omega_{-Z}\right]\left[i_{-X}\right]\left[(\nu_☾ + \omega_☾)_{-Z}\right]\begin{bmatrix} R_☾ \\ 0 \\ 0 \end{bmatrix}$$

Sun position vector (ECI)

$$\left[\bar{R}_\Theta\right] = \left[\varepsilon_{-X}\right]\left[(\nu_\Theta + \omega_\Theta)_{-Z}\right]\begin{bmatrix} R_\Theta \\ 0 \\ 0 \end{bmatrix}$$

where

$\nu_{\leftmoon}$ = The true anomaly of the moon computed from a literal expansion in terms of the sine function of the mean anomaly, M and the eccentricity to the seventh degree;

$\omega_{\leftmoon}$ = The argument of perigee of the moon's orbit;

and similarly for $\nu_{\odot}$ and $\omega_{\odot}$.

These elements are determined by the following process:

$$M_{\leftmoon} = \leftmoon - \Gamma'$$

$$M_{\odot} = L - \Gamma$$

Thence, for each respective orbit,

$$\nu = M + \left(2e - \frac{1}{4} e^3 + \frac{5}{96} e^5 + \frac{107}{4608} e^7\right) \mathrm{Sin}(M)$$

$$+ \left(\frac{5}{4} e^2 = \frac{11}{24} e^4 + \frac{17}{192} e^6\right) \mathrm{Sin}(2M)$$

$$+ \left(\frac{13}{12} e^3 - \frac{43}{64} e^5 + \frac{95}{512} e^7\right) \mathrm{Sin}(3M)$$

$$+ \left(\frac{103}{96} e^4 - \frac{451}{480} e^6\right) \mathrm{Sin}(4M)$$

Furthermore, the arguments of perigee are determined,

$$\omega_{\leftmoon} = \Gamma' - \Omega$$

$$\omega_{\odot} = \Gamma'$$

and finally the radius vectors for the sun and the moon are computed,

$$r = \frac{a(1 - e^2)}{1 + e\mathrm{Cos}\nu}$$

where the orbit mean distances, a, are obtained from the respective horizontal parallaxes,

$$a_{\leftmoon} = R_E/\Pi_{\leftmoon}$$

$$a_{\odot} = R_E/\Pi_{\odot}$$

$$a_{\leftmoon} = 384421.87 \text{ km}$$

$$a_{\odot} = 149605590. \text{ km}$$

COMPUTE SOLAR/LUNAR EPHEMERIS (EPHEM)

Calculate Lunar Ephemeris
Calculate Solar Ephemeris
Calculate Ellipses Properties
Transform Axes

Figure A-25

```
0001          SUBROUTINE EPHEM (T,RM,RO)
        C****************************************************************************
        C ************  LUNAR SOLAR EPHEMERIS DATA ************
        C
        C       INPUT PARAMETER
        C             T = TIME IN JULIAN DAYS SINCE JD 0.0
        C       OUTPUT PARAMETERS
        C             RO = RADIUS VECTOR (EARTH CENTER TO SUN)  -- ECI COORDINATES (KM)
        C             RM = RADIUS VECTOR (EARTH CENTER TO MOON) -- ECI COORDINATES (KM)
        C****************************************************************************
0002          INCLUDE 'DEBUG.COM'
00100   0003  *      COMMON /DEBUG/ IENTER,IDEBUG
00200         * C
00300         * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400         * C
00500         * C                   IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600         * C                   IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
00700         * C

        0004         DIMENSION RO(3),RM(3)
        0005         REAL*8  AM,AO,C ,CIM,CO,CVW,D,EM,EO,EP,GM,GO,LM,LO,MM,MO,OM,RMS,
                    *        ROS,SE,SIM,SO,SVW,T,T1,T2,VM,VO,VWM,VWS,RM,RO
        0006         DATA CIM,SIM/.995963,-.089763/
        0007         DATA AM,AO,EM/384421.87,1.4960559E8,.054900489/
              C
              C        D = .0001 X DAYS SINCE 1900
              C
        0008         D=1.E-4*(T-2415020.)
              C ******   MOON EPHEM RIS
              C REFERENCE FOR EQUATIONS LM AND OM BELOW, P. 107 EXPLANATORY SUPPLEMENT
        0009         LM=4.7199665716 6+D*(2299.7150294-D*(1.48352986E-6-D*6.806784E-10))
        0010         GM=5.8351515389+D*(19.44368-D*(1.3507103E-5+D*4.537856E-9))
        0011         OM=-4.52360151484+D*(9.2422029423-D*(2.71747764E-6+D*8.72665E-10))
              C ******  SUN EPHEMERIS
        0012         LO= 4.88162793797+D*(1.7202791266E+2+D*3.9566614E-7)
        0013         GO=4.90822946103+D*(8.2149855364E-3+D*(5.9166662E-7+D*1.22173E-9))
        0014         EO=0.01675104-D*(1.1444E-5+D*9.4E-9)
        0015         EP=-0.40931974744+D*(6.217909993E-5+D*(2.146755E-9-D*1.79769E-10))
              C ******  ELLIPSES PROPERTIES
        0016         MM=LM-GM
        0017         CALL TRUEA(MM,EM,VM)
        0018         RMS=AM*(1.-EM*EM)/(1.+EM*COS(VM))
        0019         MO=LO-GO
        0020         CALL TRUEA(MO,EO,VO)
        0021         ROS=AO*(1.-EO*EO)/(1.+EO*COS(VO))
              C ****** AXES TRANSFORMATION
        0022         VWM=-GM-OM-VM
        0023         CVW = COS(VWM)
        0024         SVW = SIN(VWM)
        0025         CO  = COS(OM)
        0026         SO  = SIN(OM)
        0027         CE  = COS(EP)
        0028         SE  = SIN(EP)
              C ****** TRANSFORM FROM LOCAL TO ECI COORDINATES
              C ****** MOON
        0029         RM(1) = RMS*(CVW*CO-CIM*SVW*SO)
        0030         T1 = -RMS*(CVW*SO+CIM*SVW*CO)
```

A-153

```
      T2 = RMS*SIM*SVW
      RM(2) = T1*CE+T2*SE
      RM(3) = T2*CE-T1*SE
C ****** SUN
      VWS=-(VO+GO)
      CVW = COS(VWS)
      SVW = SIN(VWS)
      RO(1) = ROS*CVW
      RO(2) = -ROS*CE*SVW
      RO(3) = ROS*SE*SVW
      RETURN
      END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	828	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	304	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		EPHEM

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	R*8	AM	2-00000008	R*8	AO	2-00000010	R*8	CE	2-00000018	R*8	CIM
2-00000020	R*8	CO	2-00000028	R*8	CVW	2-00000030	R*8	D	2-00000038	R*8	EM
2-00000040	R*8	EO	2-00000048	R*8	EP	2-00000050	R*8	GM	2-00000058	R*8	GO
3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-00000060	R*8	LM	2-00000068	R*8	LO
2-00000070	R*8	MM	2-00000078	R*8	MO	2-00000080	R*8	OM	2-00000088	R*8	RMS
2-00000090	R*8	ROS	2-00000098	R*8	SE	2-000000A0	R*8	SIM	2-000000A8	R*8	SO
2-000000B0	R*8	SVW	AP-00000004@	R*8	T	2-000000B8	R*8	T1	2-000000C0	R*8	T2
2-000000C8	R*8	VM	2-000000D0	R*8	VO	2-000000D8	R*8	VWM	2-000000E0	R*8	VWS

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	RM	24	(3)
AP-0000000C@	R*8	RO	24	(3)

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DCOS MTH$DSIN TRUEA

Total Space Allocated = 1140 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          2.07 seconds
  Elapsed Time:      16.69 seconds
  Page Faults:       363
  Dynamic Memory:    160 pages
```

```
0001        SUBROUTINE TRUEA(M,E,V)
      C*****************************************************************
      C       THIS ROUTINE COMPUTES THE TRUE ANOMALY
      C*****************************************************************
0002          REAL*8 E,E2,E3,E4,E5,E6,E7,M,S2M,S3M,S4M,SM,V
0003        E2=E*E
0004        E3=E*E2
0005        E4=E*E3
0006        E5=E*E4
0007        E6=E*E5
0008        E7=E*E6
0009        SM=SIN(M)
0010        S2M=SIN(2.*M)
0011        S3M=SIN(3.*M)
0012        S4M=SIN(4.*M)
0013        V=M+SM*(2.*E-.25*E3+0.052083333*E5+0.023220486*E7)+S2M*(1.25*E2
          1 -0.458333333*E4+0.088541667*E6)+S3M*(1.08333333*E3-0.671875*E5+
          2 0.185546875*E7)+S4M*(1.072916667*E4-0.939583333*E6)
0014         RETURN
0015         END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	311	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	80	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD

ENTRY POINTS

Address	Type	Name
0-00000000		TRUEA

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
AP-00000008@	R*8	E	2-00000000	R*8	E2	2-00000008	R*8	E3	2-00000010	R*8	E4
2-00000018	R*8	E5	2-00000020	R*8	E6	2-00000028	R*8	E7	AP-00000004@	R*8	M
2-00000030	R*8	S2M	2-00000038	R*8	S3M	2-00000040	R*8	S4M	2-00000048	R*8	SM
AP-0000000C@	R*8	V									

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DSIN

Total Space Allocated = 391 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          0.94 seconds
Elapsed Time:      7.78 seconds
Page Faults:       297
Dynamic Memory:    160 pages
```

2.5.1.2 COMPUTE SOLAR PRESSURE MODULE (SPRESS)

Solar radiation pressure is an external force acting on the spacecraft and must be included in the equations of motions. This force is calculated with the assumption that the satellite is symmetric, the photon flux is constant, and the sun behaves as a point source of photons. If the sun is occulted by either the moon or the earth, the acceleration is zero.

The acceleration of the spacecraft caused by solar radiation impinging on the exposed vehicle body is equivalent to the change in momentum of the incident radiation, divided by the mass of the spacecraft. The change in momentum is equal to the product of the momentum per photon P_o, the photon rate persecond per unit area K, the exposed area projected to a flat surface in the direction of the radiation A, the reflectivity of the surface K_1, and the unit vector from the spacecraft to the sun R_o. The acceleration is the change in momentum divided by the mass of the spacecraft, m.

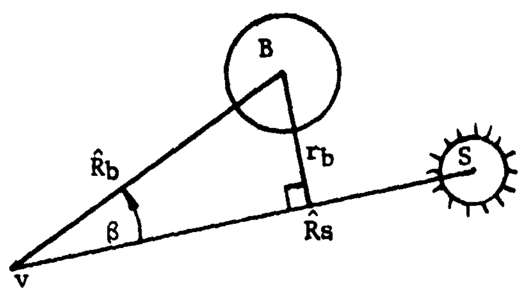
$$a_r = P_o \, K \cdot A \cdot R_s \, / \, m$$

Assuming that the spacecraft operates at a constant distance from the sun the value of K is constant and may be combined with P_o,

$$P = P_o \cdot K$$

For a spherical or near spherical spacecraft the factor (A/m) may be considered to be constant; furthermore a single value for K_1 may be used for most vehicles.

The vehicle will not experience radiation pressure from the the sun if it is haded by the earth or the moon. No shading will occur if either 1) the occulting body and the sun are on opposite hemispheres from the spacecraft, i.e., the dot products of the unit vectors from the spacecraft to these bodies is negative; or 2) the radius of the occulting body is less than the perpendicular from the spacecraft-sun direction to the center of the body.



The procedure used in this module is shown in Figure A-26 . Compute the unit vectors from the spacecraft to the sun, R_s, and to the earth, R_e, and moon, R_m (see algorithms SE-1 and ST-4). Do not compute radiation pressure if

$$R_b \cdot R_s < 0 \quad (b = \text{earth or moon})$$

or if

$$r_b < |Rb| \sin\beta$$

where

$$\beta = \cos^{-1} |Rb \cdot Rs|$$

If neither of the above conditions is met the spacecraft is irradiated by the sun and radiation pressure should be computed

$$\bar{\ddot{x}} = PK_1 \cdot \left(\frac{A}{mv}\right) \cdot Rs \qquad Km/sec^2$$

where

$$P = 4.59 \times 10^{-9} \quad \left((\text{Newtons})/m\right) \ (Km/m)$$

$$K_1 = 1.7$$

$$A/mv = \text{exposed vehicle area in } m^2/\text{vehicle mass in kg}$$

COMPUTE SOLAR RADIATION PRESSURE

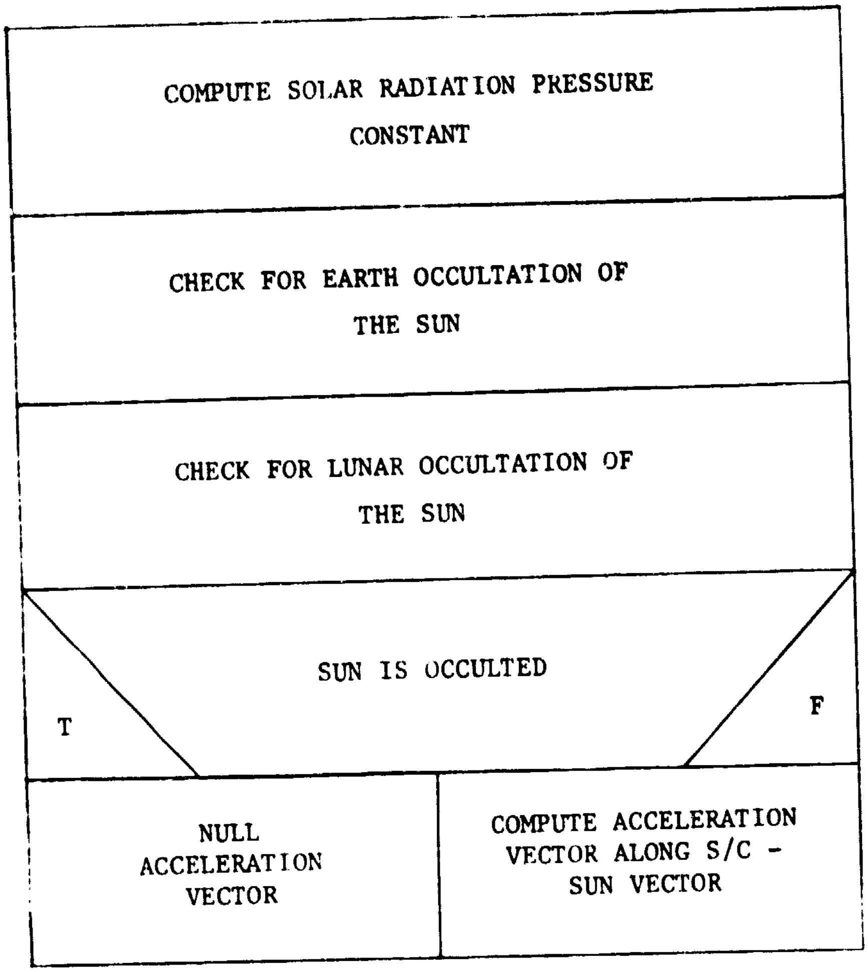


Figure A-26

```
      SUBROUTINE SPRESS(ACC)
      INCLUDE 'DEBUG.COM'
      COMMON /DEBUG/ IENTER,IDEBUG
C
C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
C
C                     I NTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
C                     IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
C
      INCLUDE 'RVEC.COM'
      COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
     .              ,RA,R2,R3,RSMA,RTG(3)
      REAL*8 R,RM,RO,RSM,RSO,RSS,SB,FA,R2,R3,RSMA,RTG
C
C               THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
C
C                     R          EARTH CENTER TO S/C -   ECI (KM)
C                     RM              "      "     MOON -   ECI (KM)
C                     RO              "      "     SUN -    ECI (KM)
C                     RSM        SPACECRAFT TO MOON -       ECI (KM)
C                     RSO             "            SUN -      ECI (KM)
C                     RSS        EARTH CENTER TO STAR -  ECI
C                     RA         ABSOLUTE OF VECTOR R (KM)
C                     R2         SQUARE OF RA (KM 2)
C                     R3         CUBE OF RA (KM 3)
C                     RSMA       ABSOLUTE OF RSM (KM)
C
      INCLUDE 'CONST.COM'
      COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
      REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
C
C               PROGRAM CONSTANTS
C
C                     ATM        'C AREA TO MASS RATIO (METERS/KG)
C                     RBM        OBSTRUCTION RADIUS OF THE MOON (KM)
C                     RBE             "          "          EARTH (KM)
C                     RBO             "          "          SUN (KM)
C                     R 2        SQUARE OF THE EARTHS RADIUS (KM 2)
C                     RM2             "        LUNAR RADIUS (KM 2)
C                     UM         LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
C                     US         SOLAR        "           "          "
C                     U_         EARTH        "           "          "
C                     J2,J3,J4   ZONAL GRAVITATIONAL HARMONIC TERMS
C                     DTU        REGULARIZED TIME STEP SIZE (SEC)
C                     PKI        SOLAR PRESSURE CONSTANT
C
      DIMENSION ACC(3)
      REAL*8 AM,FSP,ACC,VMAG
C***********************************************************************
C                 SOLAR RADIATION PRESSURE COMPUTATIONS
C***********************************************************************
      LOGICAL L
      FSP = -PK1*ATM
      CALL OCCULT(R,RSO,RBE,L)
      IF (.NOT. L) CALL OCCULT(RSM,RSO,RBM,L)
      DO 10 I=1,3
   10 ACC(I)=0.0
```

```
0018            IF(L) GO TO 40
           C**********************************************************************
           C             SOLAR RADIAN PRESSURE
           C             A = F/M = -P(A/M) K1 RSS
           C**********************************************************************
0019      20    AM = VMAG(RSO,3)
0020            DO 30 I=1,3
0021      30    ACC(I) = FSP*RSO(I)/AM
0022      40    RETURN
0023            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	112	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	4	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	96	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 RVEC	224	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 CONST	112	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		SPRESS

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	R*8	AM	5-00000000	R*8	ATM	5-00000060	R*8	DTU	2-00000008	R*8	FSP
2-00000014	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	5-00000048	R*8	J2
5-00000050	R*8	J3	5-00000058	R*8	J4	2-00000010	L*4	L	5-00000068	R*8	PK1
4-00000080	R*8	R2	4-00000088	R*8	R3	4-000000A8	R*8	RA	5-00000010	R*8	RBE
5-00000008	R*8	RBM	5-00000018	R*8	RBO	5-00000020	R*8	[illegible]E2	5-00000028	R*8	RM2
4-000000C0	R*8	RSMA	5-00000040	R*8	UE	5-00000030	R*8	UM	5-00000038	R*8	US

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	ACC	24	(3)
4-00000000	R*8	R	24	(3)
4-00000018	R*8	RM	24	(3)
4-00000030	R*8	RO	24	(3)
4-00000048	R*8	RSM	24	(3)
4-00000060	R*8	RSO	24	(3)
4-0000007[illegible]	R*8	RSS	24	(3)
4-000000C8	R*8	RTG	24	(3)

```
4-00000090  R*8  SB                    24  (3)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
**	10	**	20	**	30	0-0000006B	40

FUNCTIONS AND SUBROUTINES REFERENCED

OCCULT VMAG

Total Space Allocated = 556 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:            0.93 seconds
Elapsed Time:        14.83 seconds
Page Faults:         346
Dynamic Memory:      180 pages
```

2.5.1.3 COMPUTE SOLAR/LUNAR GRAVITATIONAL PERTURBATIONS (GPERT AND FQ)

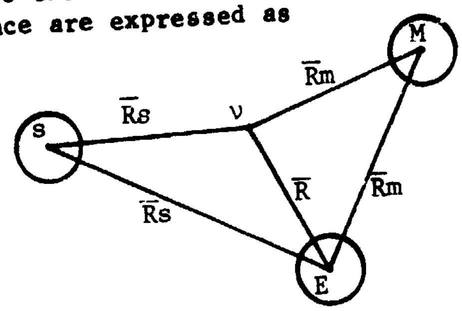
The gravitational accelerations from the moon and sun must be considered if the equations of motion are to be accurate, especially for high altitude orbits.

This algorithm computes the accelerations due to sun and moon in rectangular coordinates to be added directly to the accelerations caused by other forces. The sun and moon are considered to be point masses, and their positions as a function of time are computed in the module EPHEM.

Classically, the accelerations due to the sun and moon acting on the spacecraft in a geocentric frame of reference are expressed as

$$\bar{a} = \mu s \left[\frac{\bar{R}s}{|\bar{R}s|^3} - \frac{\bar{R}s}{|\bar{R}s|^3}\right]$$

$$\bar{a}m = \mu m \left[\frac{\bar{R}m}{|\bar{R}m|^3} - \frac{\bar{R}\mu}{|\bar{R}\mu|^3}\right]$$



Sun-Moon-Earth-Vehicle Geometry

However, owing to the numerical difficulties that may arise in the differencing of large numbers such as Rs and Rs, a somewhat different technique is used to compute a_s and a_m. This technique is described in the following procedure.

Procedure:

Compute solar perturbations,

$$\bar{a}_s = -\frac{\mu s}{|Rs|^3}\left[\bar{R} - f(qs)\cdot\bar{R}s\right]$$

Compute lunar perturbations,

$$\bar{a}_m = -\frac{\mu m}{|Rm|^3}\left[\bar{R} - f(qm)\cdot\bar{R}\mu\right]$$

where

$$f(qs) = -qs\,\frac{(3 + 3qs + qs^2)}{1 + (1 + qs)^{3/2}}$$

$$qs = \frac{|R|^2}{|Rs|^2} - 2\,\frac{|R|}{|Rs|}\cdot \text{Cos}\,\alpha s$$

and similarly for f (q_m), q_m, and cos α_m.

COMPUTE SOLAR/LUNAR PERTURBATION

```
Do while all vector components
    ACC(SUN) = -US/(RS**3) * (R - FQS * RS)

Do while all vector components
    ACC(MOON) = UM/(RM**3) * (R - RQM * RM)

Where FQ = -Q(3 + 3Q + Q**2)/(1 + (1 + Q)**3/2)

And, Q = R**2/RS**2 - 2ABS(R/RS) COS(ALPHA)

ALPHA = R DOT RS / (ABS(R) * ABS(RS))
```

Figure A-27

```
        SUBROUTINE GPERT(X,ACC)
        DIMENSION X(6),ACC(3)
        REAL*8 X,ACC,T1,T2,VMAG
        INCLUDE 'DEBUG.COM'
 *      COMMON /DEBUG/ IENTER,IDEBUG
    C
 *  C           USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
 *  C
 *  C                 I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
 *  C                 IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
 *  C
        INCLUDE 'RVEC.COM'
 *      COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
 *     .              ,RA,R2,R3,RSMA,RTG(3)
 *      REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
 *  C
 *  C           THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
 *  C
 *  C                 R          EARTH CENTER TO S/C -    ECI (KM)
 *  C                 RM            "       "      MOON -   ECI (KM)
 *  C                 RO            "       "      SUN -    ECI (KM)
 *  C                 RSM        SPACECRAFT TO MOON -       ECI (KM)
 *  C                 RSO             "         SUN -         ECI (KM)
 *  C                 RSS        EARTH CENTER TO STAR -  ECI
 *  C                 RA         ABSOLUTE OF VECTOR R (KM)
 *  C                 R2         SQUARE OF RA (KM 2)
 *  C                 R3         CUBE OF RA (KM 3)
 *  C                 RSMA       ABSOLUTE OF RSM (KM)
 *  C
        INCLUDE 'CONST.COM'
 *      COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
 *      REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
 *  C
 *  C           PROGRAM CONSTANTS
 *  C
 *  C                 ATM        S/C AREA TO MASS RATIO (METERS/KG)
 *  C                 RBM        OBSTRUCTION RADIUS OF THE MOON (KM)
 *  C                 RBE           "           "          EARTH (KM)
 *  C                 RBO           "           "          SUN (KM)
 *  C                 R 2        SQUARE OF THE EARTHS RADIUS (KM 2)
 *  C                 RM2           "          LUNAR RADIUS (KM 2)
 *  C                 UM         LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
 *  C                 US         SOLAR        "           "          "
 *  C                 U_         EARTH        "           "          "
 *  C                 J2,J3,J4   ZONAL GRAVITATIONAL HARMONIC TERMS
 *  C                 DTU        REGULARIZED TIME STEP SIZE (SEC)
 *  C                 PKI        SOLAR PRESSURE CONSTANT
 *  C
    C
    C
    C     GRAVITY PERTURBATIONS FOR HANSS
    C                 EQ-15
    C     ACC - PERTURBATION ACCELERATION VECTOR (ECI)
    C
        T1 = -US/VMAG(RSO,3)**3
        T2 = -T1*FQ(X,RO)
        DO 10 I=1,3
```

```
        10      ACC(I) = X(I)*T1+RO(I)*T2
                T1 = -UM/VMAG(RSM,3)**3
                T2 = -T1*FQ(X,RM)
                DO 20 I=1,3
        20      ACC(I) = ACC(I)+X(I)*T1+RM(I)*T2
                RETURN
                END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	213	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	4	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	108	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 RVEC	224	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 CONST	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GPERT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-00000000	R*8	ATM	5-00000060	R*8	DTU	2-00000010	I*4	I	3-00000004	I*4	IDEBUG
3-00000000	I*4	IENTER	5-00000048	R*8	J2	5-00000050	R*8	J3	5-00000058	R*8	J4
5-00000068	R*8	PK1	4-000000B0	R*8	R2	4-000000B8	R*8	R3	4-000000A8	R*8	RA
5-00000010	R*8	RBE	5-00000008	R*8	RBM	5-00000018	R*8	RBO	5-00000020	R*8	RE2
5-00000028	R*8	RM2	4-000000C0	R*8	RSMA	2-00000000	R*8	T1	2-00000008	R*8	T2
5-00000040	R*8	UE	5-00000030	R*8	UM	5-00000038	R*8	US			

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	ACC	24	(3)
4-00000000	R*8	R	24	(3)
4-00000018	R*8	RM	24	(3)
4-00000030	R*8	RO	24	(3)
4-00000048	R*8	RSM	24	(3)
4-00000060	R*8	RSO	24	(3)
4-00000078	R*8	RSS	24	(3)
4-000000C8	R*8	RTG	24	(3)
4-00000090	R*8	SB	24	(3)
AP-00000004@	R*8	X	48	(6)

```
LABELS

    Address   Label      Address   Label

      **      10           **      20

FUNCTIONS AND SUBROUTINES REFERENCED

FQ             VMAG

Total Space Allocated = 669 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:            1.17 seconds
  Elapsed Time:        7.89 seconds
  Page Faults:         348
  Dynamic Memory:      160 pages
```

```
0001            FUNCTION FQ(R,RS)
          C*****************************************************************
          C         QS FUNCTION FOR MOON-SUN GRAVITY PERTURBATIONS
          C*****************************************************************
0002            DIMENSION R(3),RS(3)
0003            REAL*8 CA,Q,RM,RSM,R,RS,FQ,VMAG,VDOT
0004            RM = VMAG(R,3)
0005            RSM = VMAG(RS,3)
0006            CA = VDOT(R,RS,3)/(RM*RSM)
0007            Q = RM/RSM*(RM/RSM -2.*CA)
0008            FQ = -Q*(3.+Q*(3.+Q))/(1.+(1.+Q)**1.5)
0009            RETURN
0010            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	158	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	4	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	116	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD

ENTRY POINTS

Address	Type	Name
0-00000000	R*8	FQ

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000008	R*8	CA	2-00000010	R*8	Q	2-00000018	R*8	RM	2-00000020	R*8	RSM

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	R	24	(3)
AP-00000008@	R*8	RS	24	(3)

FUNCTIONS AND SUBROUTINES REFERENCED

VDOT VMAG

Total Space Allocated = 278 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           0.70 seconds
  Elapsed Time:       11.55 seconds
  Page Faults:        332
  Dynamic Memory:     160 pages
```

2.5.1.4 COMPUTE NAVIGATION STATE PARTIALS (PDNAV)

The Position State Transition Matrix is a 6 x 6 matrix of partial derivatives which relate small changes in the position and velocity state at one time to changes in the position and velocity state at a later time.

The state transition matrix shall be calculated by integrating the linear differential equation

$$\phi_N = F\phi_N$$

with the initial condition $\phi_N(t_0, t_0)$ = Identity (6 x 6) where the non-zero elements of F are given by

$$F_{41} = -\frac{\mu}{R^3}\left[1 - 3\left(\frac{x_1}{R^2}\right)^2\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\left[35\frac{(x_1 \cdot x_3)^2}{R^2} - 5\frac{(x_1{}^2 + x_3{}^2)}{R^2} + 1\right]$$

$$F_{42} = \frac{\mu}{R^3}\left[\frac{3x_1 \cdot x_2}{R^2}\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\frac{(5x_1 \cdot x_2)}{R^2}\left[7\left(\frac{x_3}{R}\right)^2 - 1\right]$$

$$F_{43} = \frac{\mu}{R^3}\left[\frac{3x_1 \cdot x_3}{R^2}\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\frac{(5x_1 \cdot x_2)}{R^2}\left[7\left(\frac{x_3}{R}\right)^2 - 3\right]$$

$$F_{51} = F_{42}$$

$$F_{52} = -\frac{\mu}{R^3}\left[1 - 3\left(\frac{x_2}{R}\right)^2\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\left[35\frac{(x_2 \cdot x_3)}{R^2} - 5\frac{(x_2{}^2 + x_3{}^2)}{R^2} + 1\right]$$

$$F_{53} = \frac{\mu}{R^3}\left[\frac{3x_2 \cdot x_3}{R^2}\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\frac{(5x_2 \cdot x_3)}{R^2}\left[7\left(\frac{x_3}{R}\right)^2 - 3\right]$$

$$F_{61} = F_{43}$$

$$F_{62} = F_{53}$$

$$F_{63} = -\frac{\mu}{R^3}\left[1 - 3\left(\frac{x_3}{R}\right)^2\right] - \frac{3J_2}{2}\left(\frac{Re}{R}\right)^2 \cdot \frac{\mu}{R^3}\left[35\left(\frac{x_3}{R}\right)^4 - 30\left(\frac{x_3}{R}\right)^2 + 3\right]$$

where

R_e = equatorial radius of earth

COMPUTE NAVIGATION STATE PARTIALS

COMPUTE STATE PARTIALS - 2 ZONAL HAR- MONICS
COMPUTE THE TRANSITION MATRIX DERIVATIVES
BY MULTIPLYING TRANSITION MATRIX BY STATE PARTIALS MATRIX

Figure A-28

```
0001            SUBROUTINE PDNAV(PN,PDN)
0002            DIMENSION PN(6,6),PDN(6,6)
0003            REAL*8 F1,F2,X2R,XYR,XZR,Y2R,YZR,Z2R,PDN,PN
0004            INCLUDE 'DEBUG.COM'
00100    0005   *       COMMON /DEBUG/ IENTER,IDEBUG
00200           * C
00300           * C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400           * C
00500           * C                       IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600           * C                       IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
00700           * C
         0006           INCLUDE 'CONST.COM'
         0007   *       COMMON /CONST/ ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
         0008   *       REAL*8         ATM,RBM,RBE,RBO,RE2,RM2,UM,US,UE,J2,J3,J4,DTU,PK1
                * C
                * C               PROGRAM CONSTANTS
                * C
                * C                       ATM       S/C AREA TO MASS RATIO (METERS/KG)
                * C                       RBM       OBSTRUCTION RADIUS OF THE MOON (KM)
                * C                       RBE            "          "          EARTH (KM)
                * C                       RBO            "          "          SUN (KM)
                * C                       R 2       SQUARE OF THE EARTHS RADIUS (KM 2)
                * C                       RM2         "         LUNAR RADIUS (KM 2)
                * C                       UM        LUNAR GRAVITATION CONSTANT (KM 3/SEC 2)
                * C                       US        SOLAR        "          "          "
                * C                       U         EARTH        "          "          "
                * C                       J2,J3,J4  ZONAL GRAVITATIONAL HARMONIC TERMS
                * C                       DTU       REGULARIZED TIME STEP SIZE (SEC)
                * C                       PKI       SOLAR PRESSURE CONSTANT
                * C
         0009           INCLUDE 'RVEC.COM'
         0010   *       COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
                *     .              ,RA,R2,R3,RSMA,RTG(3)
         0011   *       REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
                * C
                * C               THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
                * C
                * C                       R         EARTH CENTER TO S/C -   ECI (KM)
                * C                       RM          "      "      MOON -  ECI (KM)
                * C                       RO          "      "      SUN -   ECI (KM)
                * C                       RSM       SPACECRAFT TO MOON -       ECI (KM)
                * C                       RSO            "         SUN -        ECI (KM)
                * C                       RSS       EARTH CENTER TO STAR -  ECI
                * C                       RA        ABSOLUTE OF VECTOR R (KM)
                * C                       R2        SQUARE OF RA (KM 2)
                * C                       R3        CUBE OF RA (KM 3)
                * C                       RSMA      ABSOLUTE OF RSM (KM)
                * C
         0012           INCLUDE 'GFPART.COM'
         0013   *       COMMON/GFPART/ FA(4,4),EA(4,12),FN(6,6)
         0014   *       REAL*8 FA,EA,FN
                * C
                * C               MEASUREMENT AND STATE PARTIALS
                * C
                * C                       FA        ATTITUDE STATE PARTIALS
                * C                       EA             "    CONSIDERED PARAMETERS PARTIALS
                * C                       FN             "    STATE PARTIALS
```

```
C
      INCLUDE 'NSTATE.COM'
C
      COMMON /NSTATE/ XD(6),X(6),RADM,RADE
      REAL*8 XD,X,RADM,RADE
C
C              POSITION STATE AND CONSIDERED PARAMETERS
C
C                   XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
C                   X         STATE POSITION PARAMETERS (KM AND KM/SEC)
C                   RADM      RADIUS OF THE MOON (KM)
C                   RADE      EARTH DETECTABLE RADIUS (KM)
C
C
      INCLUDE 'ARRAYS.COM'
C
      COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
     .               ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
      REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
C
C              THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
C
C                   T1 - T4      SINGLE DIMENSION ARRAYS
C                   T11 - T77    DUAL DIMENSIONED ARRAYS
C                   T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
C
      INCLUDE 'UPDT.COM'
C
      COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
      REAL*8 QN,QA,Q,QDOT
C
C              STATE  STIMATION PARAMETERS
C
C                   QN        NAV. DYN. NOISE COVARIANCE DIAGONAL
C                   QA        MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
C                   Q         CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
C                   QDOT      DIFFERENTIAL OF Q
C
      F1 = UE/R3
      F2 = 1.5*F1*J2*RE2/R2
      X2R = X(1)*X(1)/R2
      Y2R = X(2)*X(2)/R2
      Z2R = X(3)*X(3)/R2
      XYR = X(1)*X(2)/R2
      XZR = X(1)*X(3)/R2
      YZR = X(2)*X(3)/R2
      FN(4,1) = F1*(3.*X2R-1.)+F2*(5.*(Z2R+X2R-7.*X2R*Z2R)-1.)
      FN(5,2) = F1*(3.*Y2R-1.)+F2*(5.*(Z2R+Y2R-7.*Y2R*Z2R)-1.)
      FN(6,3) = F1*(3.*Z2R-1.)+F2*(5.*(6.*Z2R -7.*Z2R*Z2R)-3.)
      FN(4,2) = 3.*F1*XYR+5.*F2*XYR*(1.-7.*Z2R)
      FN(4,3) = 3.*F1*XZR+5.*F2*XZR*(3.-7.*Z2R)
      FN(5,3) = 3.*F1*YZR+5.*F2*YZR*(3.-7.*Z2R)
      FN(5,1) = FN(4,2)
      FN(6,1) = FN(4,3)
      FN(6,2) = FN(5,3)
      CALL MATAB(FN,PN,PDN,6,6,6)
C*************************************************************************
```

```
C COMPUTE THE DIFFERENTIAL FOR CONTRIBUTION TO COVN DUE TO DYNAMIC NOISE
C I.E COMPUTE QDOT = FN*Q + Q*FN(TRANSPOSE) + QN
C                  = FN*Q + QN + (FN*Q)TRANSPOSE
C**********************************************************************
      CALL MATAB(FN,Q,QDOT,6,6,6)
      DO 200 I = 1,6
          DO 200 J = 1,6
              QDOT(I,J) = QDOT(I,J) + QDOT(J,I)
              IF (I.EQ.J) GO TO 205
                          GO TO 210
205           QDOT(I,J) = QDOT(I,J) + QN(I)
              GO TO 200
210           QDOT(J,I) = QDOT(I,J)
200           CONTINU
      IF (IDEBUG.GT.3) WRITE(6,900) F1,F2,((FN(I,J),J=1,6),I=1,6),
     . ((PDN(I,J),J=1,6),I=1,6),((QDOT(I,J),J=1,6),I=1,6)
900   FORMAT(4X,' IN PDNAV:',/,
     . 10X,' F1 = ',E17.10,' F2 = ',E17.10,
     . /,10X,' FN:',/,6(10X,6(1X,E17.10),/),
     . 10X,' PDN:',/,6(10X,6(1X,E17.10),/),
     . 10X,' QDOT:',/,6(10X,6(1X,E17.10),/))
      RETURN
      END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	682	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	126	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	176	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	CONST	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	RVEC	224	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	GFPART	800	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	ARRAYS	1096	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
9	UPDT	752	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		PDNAV

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
4-00000000	R*8	ATM	4-00000060	R*8	DTU	2-00000000	R*8	F1	2-00000008	R*8	F2
2-00000040	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-00000044	I*4	J
4-00000048	R*8	J2	4-00000050	R*8	J3	4-00000058	R*8	J4	4-00000068	R*8	PK1

5-00000080	R*8	R2	5-00000088	R*8	R3	5-000000A8	R*8	RA	7-00000068	R*8	RADE
7-00000060	R*8	RADM	4-00000010	R*8	RBE	4-00000008	R*8	RBM	4-00000018	R*8	RBO
4-00000020	R*8	RE2	4-00000028	R*8	RM2	5-000000C0	R*8	RSMA	4-00000040	R*8	UE
4-00000030	R*8	UM	4-00000038	R*8	US	2-00000010	R*8	X2R	2-00000018	R*8	XYR
2-00000020	R*8	XZR	2-00000028	R*8	Y2R	2-00000030	R*8	YZR	2-00000038	R*8	Z2R

ARRAYS

Address	Type	Name	Bytes	Dimensions
6-00000080	R*8	EA	384	(4, 12)
6-00000000	R*8	FA	128	(4, 4)
6-00000200	R*8	FN	288	(6, 6)
AP-00000008@	R*8	PDN	288	(6, 6)
AP-00000004@	R*8	PN	288	(6, 6)
9-00000080	R*8	Q	288	(6, 6)
9-00000030	R*8	QA	128	(16)
9-00000100	R*8	QDOT	288	(6, 6)
9-00000000	R*8	QN	48	(6)
5-00000000	R*8	R	24	(3)
5-00000018	R*8	RM	24	(3)
5-00000030	R*8	RO	24	(3)
5-00000048	R*8	RSM	24	(3)
5-00000060	R*8	RSO	24	(3)
5-00000078	R*8	RSS	24	(3)
5-000000C8	R*8	RTG	24	(3)
5-00000090	R*8	SB	24	(3)
8-00000000	R*8	T1	24	(3)
8-00000098	R*8	T11	72	(3, 3)
8-00000018	R*8	T2	24	(3)
8-00000030	R*8	T3	24	(3)
8-000000E0	R*8	T33	72	(3, 3)
8-00000048	R*8	T4	80	(10)
8-00000128	R*8	T44	128	(4, 4)
8-000003E8	R*8	T5	32	(4)
8-00000408	R*8	T6	32	(4)
8-000001A8	R*8	T66	288	(6, 6)
8-00000428	R*8	T7	32	(4)
8-000002C8	R*8	T77	288	(6, 6)
7-00000030	R*8	X	48	(6)
7-00000000	R*8	XD	48	(6)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
0-000001FD	200	0-000001EA	205	0-000001F4	210	1-00000000	900'

FUNCTIONS AND SUBROUTINES REFERENCED

MATAB

Total Space Allocated = 4088 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          3.16 seconds
  Elapsed Time:      44.34 seconds
  Page Faults:       387
  Dynamic Memory:    160 pages
```

2.6 Generate Measurement (MEASURE)

MEASURE acts as an executive directing the program flow to the module appropriate for the simulation of the type of measurement to be made. The branch to the specific measurement simulation subroutine is based on an event code that may range from 1 through 15. Those significant to MEASURE are 1 through 5. The mathematics relating to specific simulations will be spoken to under that routine name. The measurement simulation performed under specific event codes are listed below.

Event Code	Measurement Simulation Selected
1	Landmark Tracker Measurement (GCP) with alternate actions
2	Global Positioning System (GPS) Measurement
3	Star Tracker Measurement with ST1
4	Star Tracker Measurement with ST2
5	No Measurements

Alternate actions are made available under event code 1 if cloud cover is found to be in excess of 40%. These actions are to make star tracker measurements with priority ST1, ST2 unless ST1 or both fixed trackers are occulted by a major body--earth, sun, or moon.

Should the cloud cover be determined to be less than 40% but greater than 10%, the accuracy of the landmark measurement is degraded from an uncertainty of 3 meters 1σ to 30 meters 1σ .

If the event code is one, indicating a GCP sighting, the cloud cover is determined by table hookup with linear interpolation driven by a random number from a random number generator.

If the cloud coverage is greater than 40% the need for alternate action is tested by examining the norm of the body attitude quaternian covariance. If this exceeds a specified bound an attitude update will be made using a star tracker if not occulted by a major body.

After determination of cloud coverage, or if the original event code was not 1, the program branches to the appropriate subroutine responsible for generating the measurement.

Figure A-29 [illegible] rates the process of MEASURE.

SUBROUTINE MEASURE

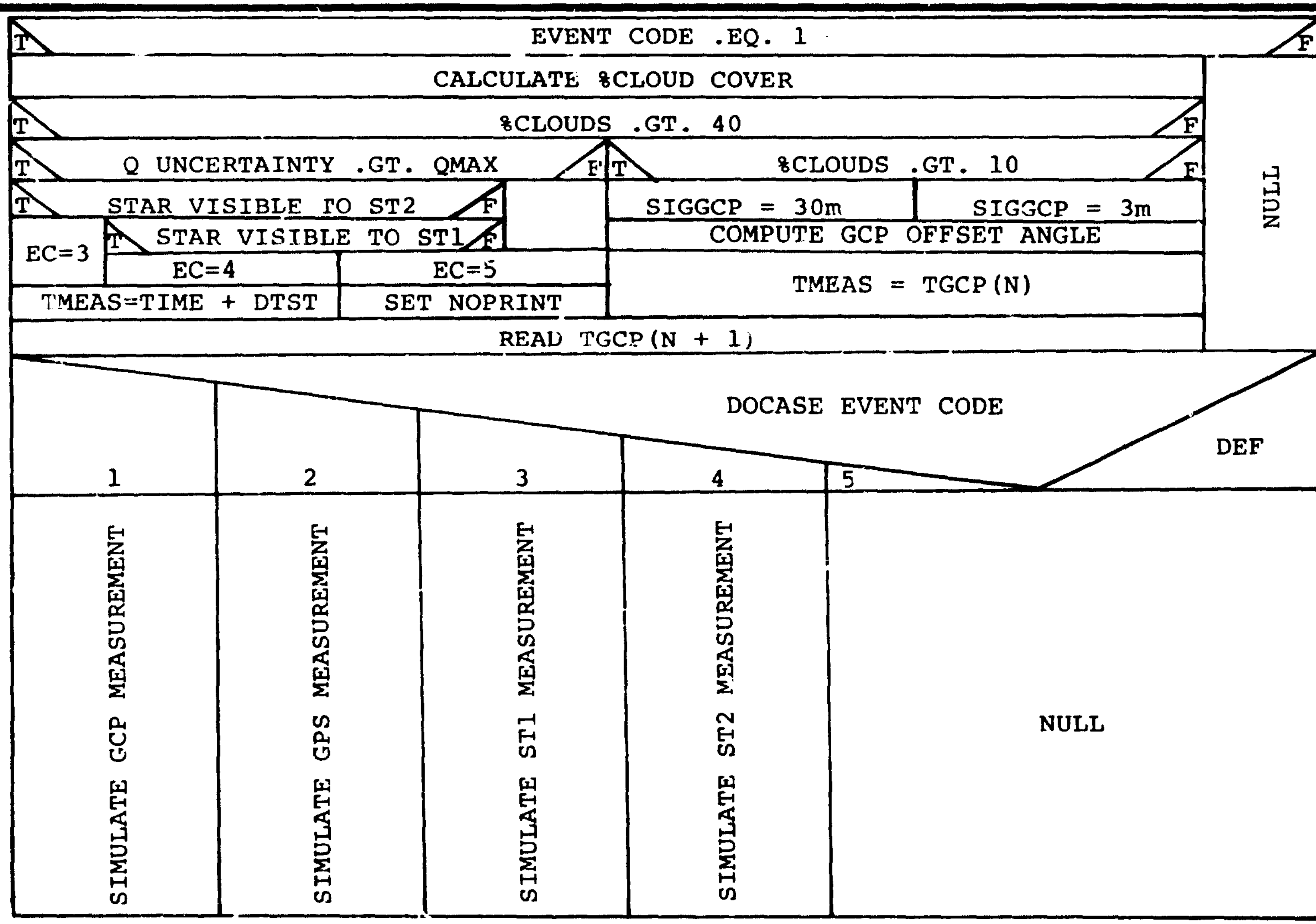


Figure A-29

```
              SUBROUTINE MEASURE
       C***********************************************************************
       C
       C        SUBROUTINE MEASURE ACTS AS THE EXECUTIVE FOR ALL SPACECRAFT
       C        MEASUREMENTS MADE USING AN EXTERNAL STIMULUS.
       C
       C        INPUT VARIABL S
       C                MCODE    = MEASUREMENT CODE - THOSE SIGNIFICANT TO
       C                                THIS MODULE ARE 1,2,3 AND 4
       C                                1 = LANDMARK TRACKER MEASUREMENT
       C                                2 = GPS MEASUREMENT
       C                                3 = STARTRACKER 1 MEASUREMENT
       C                                4 = STARTRACKER 2 MEASUREMENT
       C
       C        OUTPUT VARIABLES
       C                THE OUTPUT VARIABLES ARE EXPLICIT IN THE
       C                SUBROUTINES CALLED
       C
       C        PROGRAMMED BY JACK MYERS 13JUNE1980
       C                        EXT 4443
       C
       C***********************************************************************
                REAL*8 QNORM,PCNT,VECT1,VECT2
       C        COMMON BLOCKS
                INCLUDE 'TARG TS.COM'
     *        COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
     *        LOGICAL JFLAG
     *        REAL*8 PI,TPI
     * C
     * C                MEASUR MENT SPECIFICATIONS
     * C
     * C                        MTYPE      MEASUREMENT TYPE
     * C                        JFLAG      SET FOR STAR OBSTRUCTION
     * C                        MCODE      "    "  MEASUREMENT PROCESSING
     * C                        PI         PI
     * C                        TPI        2*PI
     * C
                INCLUDE 'DEBUG.COM'
     *        COMMON /DEBUG/ IENTER,IDEBUG
     * C
     * C                USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
     * C
     * C                        I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
     * C                        IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
     * C
                INCLUDE 'LMTPAR.COM'
     *          COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
     *       .          BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
     *          REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
     *       .          THET
     * C
     * C
     * C        LANDMARK TRACKER PARAMETERS
     * C                AL       = ALTITUDE OF LANDMARK (KM)
     * C                LON      = LONGITUDE OF LANDMARK (DEG)
     * C                LAT      = LATITUDE OF LANDMARK (DEG)
     * C                TBNL     = ORIENTATION ARRAY FOR LANDMARK TRACKER
```

```
              * C                           NOMINAL TO BODY
              * C              TNL      = MISALIGNMENT ARRAY - ACTUAL
              * C                           TRACKER TO NOMINAL
              * C              BL       = BIAS - ACTUAL (RAD)
              * C              SL       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
              * C              BKL      = BIAS - KNOWLEDGE (RAD)
              * C              THET     = LOOK ANGLE (RAD)
              * C              SKL      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
              * C              TIEO     = INITIAL EARTH FIXED TO INERTIAL
              * C                           TRANSFORMATION
              * C              TNLK     = MISALIGNMENT ARRAY KNOWLEDGE
              * C                           TRACKER TO NOMINAL
              * C              SIGGCP   = POSITION UNCERTAINTY DUE TO CLOUDS
              * C
     0012                   INCLUDE 'TIME.COM'
              * C
     0013     *           COMMON /TIME/  TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
              *          .               ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
     0014     *           REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
              *          .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
              * C
              * C              THESE ARE THE TIME REFERENCE FRAMES
              * C
              * C                   TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
              * C                   TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
              * C                   TSTOP     RUN TERMINATION TIME (SEC)
              * C                   TIA       ATTITUDE INTEGRATION TIME (SEC)
              * C                   DEL           "           "      STEP SIZE (SEC)
              * C                   TIN       POSITION INTEGRATION TIME (SEC)
              * C                   DTN           "           "      STEP SIZE (SEC)
              * C                   DATEO     DATE OF FLIGHT EPOCH (JD)
              * C                   DATER     DATE OF 1950 EPOCH (JD)
              * C                   TZERO     START TIME IN SECS. SINCE DATEO
              * C                   TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
              * C                   TIS       REAL WORLD REFERENCE TIME (SEC)
              * C                   TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
              * C                   DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
              * C                             TOO LARGE AT MEASUREMENT TIME
              * C                   TPRINT    TIME FOR PRINT (SEC)
              * C                   DTPRINT   INCREMENT ON TPRINT (SEC)
              * C
     0015                   INCLUDE 'PHIA.COM'
     0016     *           COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
              *          .      COVA(16,16),POA(16,16),QMAX
     0017     *           REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
              * C
              * C              THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
              * C
              * C                   PA        ATTITUDE STATE TRANSITION MATRIX
              * C                   TA            "      PARAMETER TRANSITION MATRIX
              * C                   PDA           "      DERIVATIVE OF TRANSITION MATRICES
              * C                   PHIA          "      AGGREGATE TRANSITION MATRIX
              * C                   COVA          "      NEW COVARIANCE MATRIX
              * C                   POA           "      PREVIOUS COVARIANCE MATRIX
              * C                   QMAX          "      COVARIANCE NORM MAX
              * C
     0018                   INCLUDE 'MODE.COM'
```

```
0019  *         COMMON /MODE/ MODE(10)
      * C
      * C               MODE(1) = LANDMARK TRACKER SWEEP MODE
      * C                         0 = RANDOM
      * C                         1 = FIXED AT INPUT THET
      * C                         2 = NO DEFAULT TO STAR TRACKER
      * C               MODE(2) = CLOUD SELECTION MODE
      * C                         0 = RANDOM CLOUD DENSITIES BASED
      * C                                 ON INPUT TABLES CLOTBL
      * C                         1 = FIXED DENSITY AT NO CLOUDS
      * C                         2 = NO CLOUDS WITH 100% CLOUD
      * C                                 COVER FOR A SPECIFIED
      * C                                 PERIOD (CLOTBL(11,12))
      * C               MODE(3-10) NOT SPECIFIED AT PRESENT
      * C
0020            DATA I1,I2/54321,12345/
0021            IF(MCODE .NE. 1) GO TO 1000
        C*************************************************************
        C       CALCULATE PERCENT CLOUD COVER
        C*************************************************************
0022            CALL RANDU(I1,I2,X)
0023            CALL CLDTBL(X,PCNT)
0024            IF(PCNT .GT. 40.) GO TO 100
0025            SIGGCP=3.
0026            IF(PCNT .GT. 10.) SIGGCP=30.
        C*************************************************************
        C       COMPUTE OFFSET ANGLE
        C*************************************************************
0027            CALL RDGCP
0028            GO TO 1000
        C*************************************************************
        C       COMPUTE NORM OF ATTITUDE QUATERNIAN COVARIANCE
        C               (AN APPROXIMATION TO VARIANCE)
        C*************************************************************
0029    100     QNORM=0.
0030            DO 105 I=1,4
0031    105     QNORM=QNORM+COVA(I,I)
0032            QNORM=SQRT(QNORM)
0033            IF(QNORM .LE. QMAX) GO TO 500
0034            IF(MODE(1) .EQ. 2) GO TO 500
        C*************************************************************
        C       CHECK TO SEE IF KNOWLEDGE OF STAR LOCATION IS VISIBLE
        C       TO ST1
        C*************************************************************
0035            CALL BVECT(VECT1,1)
0036            CALL VISIBLE(VECT1,5,L)
0037            IF(.NOT. L) GO TO 300
        C*************************************************************
        C       CHECK TO SEE IF KNOWLEDGE OF STAR LOCATION IS VISIBLE
        C       TO ST2
        C*************************************************************
0038            CALL BVECT(VECT2,2)
0039            CALL VISIBLE(VECT2,5,L)
0040            IF(.NOT. L) GO TO 200
0041    500     MCODE=5
0042            GO TO 1000
0043    200     MCODE=4
```

```
              GO TO 1000
   300        MCODE=3
   1000       CONTINUE
C*************************************************************************
C             BRANCH TO THE APPROPRIATE SUBROUTINE ON MCODE
C*************************************************************************
              GO TO (10,20,30,40),MCODE
              RETURN
C*************************************************************************
C             MAKE LANDMARK TRACKER MEASUREMENTS
C*************************************************************************
   10         CALL LAMKT
              RETURN
C*************************************************************************
C             MAKE GPS MEASUREMENTS
C*************************************************************************
   20         CALL GPS
              RETURN
C*************************************************************************
C             MAKE MEASUREM NTS WITH STARTRACKER 1
C*************************************************************************
   30         CALL START(1)
              RETURN
C*************************************************************************
C             MAKE MEASUREM_NTS WITH STARTRACKER 2
C*************************************************************************
   40         CALL START(2)
              RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	248	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	12	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	152	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 TARGETS	36	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 LMTPAR	392	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7 PHIA	7176	PIC OVR REL GBL SHR NOEXE RD WRT LONG
8 MODE	40	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		MEASURE

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-00000000	R*8	AL	6-00000038	R*8	DATEO	6-00000070	R*8	DATER	6-00000020	R*8	DEL
6-00000068	R*8	DTA	6-00000030	R*8	DTN	6-00000080	R*8	DTPRINT	2-0000002C	I*4	I
2-00000020	I*4	I1	2-00000024	I*4	I2	4-00000004	I*4	IDEBUG	4-00000000	I*4	IENTER
3-00000004	I*4	IS	3-0000000C	L*4	JFLAG	2-00000030	I*4	L	5-00000010	R*8	LAT
5-00000008	R*8	LON	3-00000010	I*4	MCODE	3-00000000	I*4	MTYPE	3-00000008	I*4	NS
2-00000008	R*8	PCNT	3-00000014	R*8	PI	7-00001C00	R*8	QMAX	2-00000000	R*8	QNORM
5-00000178	R*8	SIGGCP	5-00000180	R*8	THET	6-00000018	R*8	TIA	6-00000000	R*8	TIME
6-00000028	R*8	TIN	6-00000058	R*8	TIS	6-00000060	R*8	TISN	6-00000048	R*8	TMEAS
6-00000008	R*8	TNEXT	3-0000001C	R*8	TPI	6-00000078	R*8	TPRINT	6-00000050	R*8	TRACK
6-00000010	R*8	TSTOP	6-00000040	R*8	TZERO	2-00000010	R*8	VECT1	2-00000018	R*8	VECT2
2-00000028	R*4	X									

ARRAYS

Address	Type	Name	Bytes	Dimensions
5-000000C8	R*8	BKL	16	(2)
5-000000A8	R*8	BL	16	(2)
7-00000C00	R*8	COVA	2048	(16, 16)
8-00000000	I*4	MODE	40	(10)
7-00000000	R*8	PA	128	(4, 4)
7-00000200	R*8	PDA	512	(4, 16)
7-00000400	R*8	PHIA	2048	(16, 16)
7-00001400	R*8	POA	2048	(16, 16)
5-000000D8	R*8	SKL	16	(2)
5-000000B8	R*8	SL	16	(2)
7-00000080	R*8	TA	384	(4, 12)
5-00000018	R*8	TBNL	72	(3, 3)
5-00000130	R*8	TIEO	72	(3, 3)
5-00000060	R*8	TNL	72	(3, 3)

```
5-0000C00E8  R*8  TNLK                72  (3, 3)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-000000D4	10	0-000000DC	20	0-000000E4	30	0-000000EE	40	0-00000048	100	**	105
0-000000B3	200	0-000000BC	300	0-000000AA	500	0-000000C3	1000				

FUNCTIONS AND SUBROUTINES REFERENCED

```
BVECT      CLDTBL      FOR$RANDU    GPS      LAMKT      MTH$DSQRT      RDGCP      START
VISIBLE
```

Total Space Allocated = 8200 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /L MEASURE

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          1.99 seconds
  Elapsed Time:      20.71 seconds
  Page Faults:       171
  Dynamic Memory:    47 pages
```

2.6.1 GPS Model (GPS)

Subroutine GPS simulates a Global Positioning System measurement by corrupting the true vehicle position-velocity vector with bias and random noise. Compensation is then made for any biases known to be in the measurement. On the first pass through the subroutine, the measurement noise covariance is computed. All constraints and variables are communicated through common.

Simulation Equations

Measurements

$$\underline{MP} = \underline{P} + \underline{PB} + \underline{N}(0,PS) - \underline{PBK}$$

$$\underline{MV} = \underline{V} + \underline{VB} + \underline{N}(0,VS) - \underline{VBK}$$

where $\underline{N}(\mu,\sigma)$ = normal random variable with mean, μ, and standard deviation, σ.

Covariance

$$[RGPS] = \begin{bmatrix} PSK(1)^2 & & & & & \\ & PSK(2)^2 & & & \bigcirc & \\ & & PSK(3)^2 & & & \\ & \bigcirc & & VSK(1)^2 & & \\ & & & & VSK(2)^2 & \\ & & & & & VSK(3)^2 \end{bmatrix}$$

GPS MODEL (GPS)

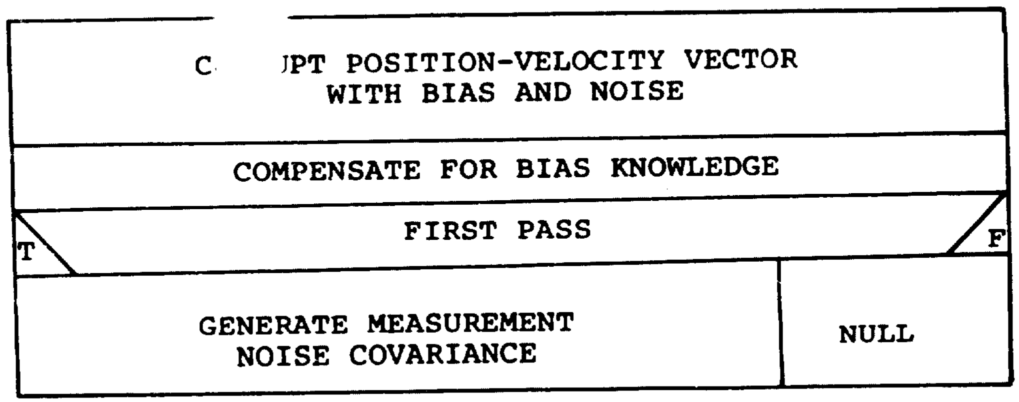


Figure A-30

```
0001                SUBROUTINE GPS
              C*************************************************************
              C
              C     SUBROUTINE GPS SIMULATES A MEASUREMENT MADE BY AN ONBOARD
              C     GPS RECEIVER BY SIMPLY ADDING BIAS AND NOISE TO THE
              C     ACTUAL VEHICL  POSITION AND VELOCITY STATE VECTOR.
              C     ONCE THE MEASUREMENT IS MADE  , COMPENSATION FOR TN HE
              C     BIAS TO THE EXTENT OF BIAS KNOWLEDGE IS SUBTRACTED
              C     FROM THE MEASUREMENT POSITION AND VELOCITY VECTOR.
              C
              C     INPUT PARAMET RS
              C             P    =   ACTUAL POSITION VECTOR (KM)
              C             V    =      "   VELOCITY   "    (KM/S)
              C             PB   =      "   POSITION BIAS VECTOR (KM)
              C             VB   =      "   VELOCITY  "     "    (KM/S)
              C             PS   =      "   POSITION NOISE STANDARD DEVIATION
              C                              (KM)
              C             VS   =      "   VELOCITY   "       "        "
              C                              (KM/S)
              C             PBK  =   POSITION BIAS KNOWLEDGE (KM)
              C             VBK  =   VELOCITY   "       "     (KM/S)
              C             PSK  =   POSITION STANDARD DEVIATION KNOWLEDGE
              C                              (KM)
              C             VSK  =   VELOCITY    "         "          "
              C                              (KM/S)
              C
              C     OUTPUT PARAMETERS
              C             MP   =   MEASUREMENT POSITION VECTOR (KM)
              C             MV   =      "        VELOCITY    "    (KM/S)
              C             RGPS=   COVARIANCE OF GPS MEASUREMENT NOISE
              C                              (KM)**2,(KM/S)**2
              C
              C     PROGRAMMED BY JACK MYERS -------- EXTENSION 4443
              C                                        4JUNE1980
              C
              C
              C*************************************************************
              C     COMMON BLOCKS
0002                INCLUDE 'ENVIR.COM'
0003      *         COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
0004      *         REAL*8 STATE,PROFILE
          *   C
          *   C             REAL WORLD STATE PARAMETERS
          *   C
          *   C                   STATE     STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
          *   C                   PROFILE   ATTITUDE PROFILE-TIME (SEC) VS
          *   C                             INERTIAL ANGULAR RATES (RAD/SEC)
          *   C                   INIT      INTEGRATION INITIALIZATION KEY (-1)
          *   C
0005                INCLUDE 'DEBUG.COM'
0006      *         COMMON /DEBUG/ IENTER,IDEBUG
          *   C
          *   C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
          *   C
          *   C                   IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
          *   C                   IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
          *   C
```

```
      INCLUDE 'GPSPAR.COM'
      COMMON /GPSPAR/ PB(3),VB(3),PS(3),VS(3),PBK(3),VBK(3),PSK(3),
     .        VSK(3)
      REAL*8 PB,VB,PS,VS,PBK,VBK,PSK,VSK
C
C     GPS PARAMETERS
C               PB      = POSITION BIAS - ACTUAL
C               VB      = VELOCITY BIAS - ACTUAL
C               PS      = POSITION NOISE STANDARD DEVIATION - ACTUAL
C               VS      = VELOCITY NOISE STANDARD DEVIATION - ACTUAL
C               PBK     = POSITION BIAS - KNOWLEDGE
C               VBK     = VELOCITY BIAS - KNOWLEDGE
C               PSK     = POSITION NOISE STANDARD DEVIATION -
C                               KNOWLEDGE
C               VSK     = VELOSITY NOISE STANDARD DEVIATION -
C                               KNOWLEDGE
C
      INCLUDE 'MEASOUT.COM'
      COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
     .        RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
     .        EDHS(2),EDVS(2),EDHL,EDVL
      REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
     .        EDHS,EDVS,EDHL,EDVL
C
C
C     MEASUREMENT OUTPUT PARAMETERS
C               MX      = POSITION/VELOCITY STATE MEASUREMENT - GPS
C                               (KM,KM/SEC)
C               EMXG    = ESTIMATED POSITION/VELOCITY STATE
C                               MEASUREMENT - HGPS
C               RGPS    = STATE MEASUREMENT NOISE COVARIANCE
C                               (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
C               DHCS    = STAR MEASUREMENT HORIZONTAL DEVIATION
C                               FROM BORESIGHT - START (RAD)
C               DVCS    = STAR MEASUREMENT VERTICAL DEVIATION
C                               FROM BORESIGHT - START (RAD)
C               EDHS    = ESTIMATED STAR MEASUREMENT HORIZONTAL
C                               DEVIATION FROM BORESIGHT (RAD)
C               EDVS    = ESTIMATED STAR MEASUREMENT VERTICAL
C                               DEVIATION FROM BORESIGHT (RAD)
C               MS      = STAR MEASUREMENT UNIT VECTOR (SECOND
C                               SUBSCRIPT REFERS TO TRACKER) - START
C               RS      = STAR MEASUREMENT NOISE COVARIANCE
C                               (KNOWLEDGE) - START (RAD**2)
C               DHCL    = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
C                               FROM BORESIGHT - LAMKT (RAD)
C               DVCL    = LANDMARK MEASUREMENT VERTICAL DEVIATION
C                               FROM BORESIGHT - LAMKT (RAD)
C               EDHL    = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
C                               DEVIATION FROM BORESIGHT (RAD)
C               EDVL    = ESTIMATED LANDMARK MEASUREMENT VERTICAL
C                               DEVIATION FROM BORESIGHT (RAD)
C               LMU     = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
C               RL      = LANDMARK MEASUREMENT NOISE COVARIANCE
C                               (KNOWLEDGE) - LAMKT (RAD**2)
C
C
```

```
            C        EQUIVALENCE
    0013             EQUIVALENCE (MX,MP),(MX(4),MV),(P,STATE),(V,STATE(4))
            C
            C        DIMENSION
    0014             DIMENSION MP(3),MV(3),P(3),V(3)
            C
    0015             REAL*8 MP,MV,V,P,GAUSS
    0016             DATA IFLAG /0/
            C*************************************************************
            C        ADD BIAS AND NOISE TO THE POSITION/VELOCITY VECTOR
            C        THEN SUBTRACT BIAS KNOWLEDGE
            C*************************************************************
    0017             DO 10 I=1,3
    0018             MP(I)=P(I)+PB(I)+GAUSS(0.,PS(I))-PBK(I)
    0019      10     MV(I)=V(I)+VB(I)+GAUSS(0.,VS(I))-VBK(I)
    0020             IF (IFLAG .EQ. 1) RETURN
            C*************************************************************
            C        SET UP NOISE COVARIANCE - FIRST PASS ONLY
            C*************************************************************
    0021             IFLAG=1
    0022             DO 15 I=1,6
    0023             DO 15 J=1,6
    0024      15     RGPS(I,J)=0.
    0025             DO 20 I=1,3
    0026             RGPS(I,I)=PSK(I)*PSK(I)
    0027      20     RGPS(I+3,I+3)=VSK(I)*VSK(I)
    0028             RETURN
    0029             END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	210	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	4	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	24	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	LONG
3 ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 GPSPAR	192	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6 MEASOUT	648	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GPS

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
6-000001E0	R*8	DHCL	6-000001E8	R*8	DVCL	6-00000278	R*8	EDHL	6-00000280	R*8	EDVL
2-00000004	I*4	I	4-00000004	I*4	IDEBUG	4-00000000	I*4	IENTER	2-00000000	I*4	IFLAG
3-00000190	I*4	INIT	2-00000006	I*4	J						

ARRAYS

Address	Type	Name	Bytes	Dimensions
6-00000150	R*8	DHCS	16	(2)
6-00000160	R*8	DVCS	16	(2)
6-00000258	R*8	EDHS	16	(2)
6-00000268	R*8	EDVS	16	(2)
6-00000228	R*8	EMXG	48	(6)
6-000001F0	R*8	LMU	24	(3)
6-00000000	R*8	MP	24	(3)
6-00000170	R*8	MS	48	(3, 2)
6-00000018	R*8	MV	[illegible]	(3)
6-00000000	R*8	MX	[illegible]	(6)
3-00000000	R*8	P	[illegible]	(3)
5-00000000	R*8	PB	24	(3)
5-00000060	R*8	PBK	24	(3)
5-00000050	R*8	PROFILE	320	(10, 4)
5-00000030	R*8	PS	24	(3)
5-00000090	R*8	PSK	24	(3)
6-00000030	R*8	RGPS	288	(6, 6)
6-00000208	R*8	RL	32	(2, 2)
6-000001A0	R*8	RS	64	(2, 2, 2)
3-00000000	R*8	STATE	80	(10)
3-00000018	R*8	[illegible]	24	(3)
5-00000018	R*8	VB	24	(3)
5-00000078	R*8	VBK	24	(3)
5-00000048	R*8	VS	24	(3)

```
5-000000A8  R*8  VSK                    24  (3)
```

LABELS

Address	Label	Address	Label	Address	Label
**	10	**	15	**	20

FUNCTIONS AND SUBROUTINES REFERENCED

GAUSS

Total Space Allocated = 1490 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

A-192

COMPILATION STATISTICS

```
Run Time:         1.42 seconds
Elapsed Time:     21.19 seconds
Page Faults:      358
Dynamic Memory:   160 pages
```

2.6.2 Star Tracker Model (START)

Subroutine START simulates a star tracker measurement made by the Kth tracker, where K is passed through the subroutine calling sequence. For purposes of simulation the star of interest is assumed to be along the Kth tracker boresight. A unit vector along the Kth tracker boresight is established and corrupted with bias and noise. Compensation for known bias is made by subtracting this knowledge from the unit vector. After the corrupted vector is unitized, the data is passed out through common. The measurement noise covariance is reestablished on each pass. This information is also passed out through common.

GAUSS is required to generate normally distributed random measurement noise.

Mathematical Specification -

Establish a unit vector along the tracker boresight.

$$\underline{U} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Establish star measurement vector by adding bias and noise to the unit vector, and compensating for bias knowledge.

$$\underline{MS}(K) = \underline{U} + \underline{BS}(K) + \underline{N}(0,SS(K)) - BSK(K)$$

$\underline{N}(\mu,\sigma)$ = Normal variable with mean, μ , and standard deviation, σ .

Reconstruct a unit vector

Establish sensor measurement

DVCS = ASIN(MS(1,K)
DHCS = ASIN(MS(2,K)/COS(DVCS(K)))

Establish measurement covariance

$$\left[RS(K)\right] = \begin{bmatrix} SSK(1,K)^2 & 0 \\ 0 & SSK(2,K)^2 \end{bmatrix}$$

STAR TRACKER MODEL (START)

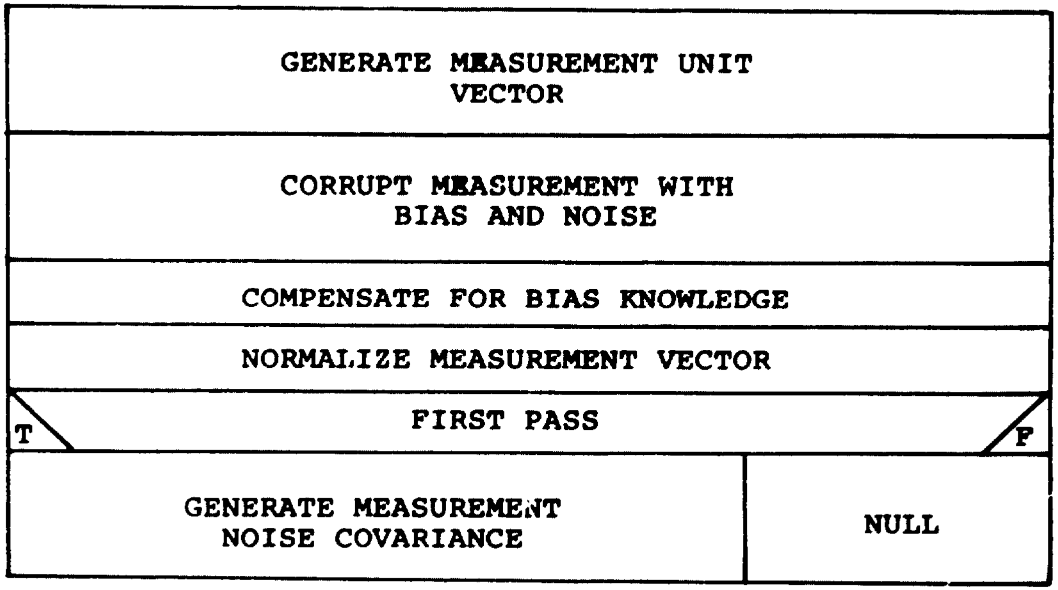


Figure A-31

```
0001          SUBROUTINE START(K)
        C*****************************************************************
        C
        C     SUBROUTINE START SIMULATES A MEASUREMENT MADE BY AN
        C     ONBOARD FIXED STAR TRACKER BY ADDING BIAS AND NOISE
        C     TO A UNIT VECTOR ALONG THE STAR TRACKER BORESIGHT.
        C     ONCE THE MEASUREMENT IS MADE COMPENSATION FOR THE KNOWN
        C     BIAS IS MADE BY SUBTRACTING THAT BIAS FROM THE
        C     MEASUREMENT
        C
        C     INPUT PARAMET RS
        C            Q      = INERTIAL TO BODY QUATERNIANS (4*1)
        C            BS     = STAR TRACKER ACTUAL BIAS VECTOR (2*1)
        C                             (RAD)
        C            SS     =    "       "       "    NOISE STANDARD
        C                             (RAD)
        C                             DEVIATION (2*1)
        C            TNS    =    "       "     MISALIGNMENT ARRAY (3*3)
        C                             (STAR TRACKER TO NOMINAL)
        C            TBNS   =    "       "     ORIENTATION ARRAY (3*3)
        C                             (NOMINAL TO BODY)
        C            BSK    =    "       "     BIAS KNOWLEDGE (3*3)
        C                             (RAD)
        C            SSK    =    "       "     NOISE STANDARD DEVIATION
        C                             KNOWLEDGE (2*1) (RAD)
        C            TNSK   =    "       "     MISALIGNMENT KNOWLEDGE
        C                             (3*3)(STAR TRACKER TO NOMINAL)
        C
        C     OUTPUT VARIABLES
        C            DHCS   = HORIZONTAL DEVIATION FROM BORESIGHT (RAD)
        C            DVCS   = VERTICAL DEVIATION FROM BORESIGHT (RAD)
        C            MS     = MEASUREMENT STAR VECTOR (3*1)
        C            RS     = STAR NOISE COVARIANCE (2*2)    CHECK
        C                             (RAD)**2
        C
        C
        C     PROGRAMMED BY JACK MYERS 11JUNE1980
        C                   EXT 4443
        C
        C*****************************************************************
        C     COMMON BLOCKS
        C     INCLUDE 'DEBUG.COM'
0002
0003    *     COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C           USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                 IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                 IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
0004          INCLUDE 'STARPAR.COM'
0005    *     COMMON /STARPAR/ BS(2,2),SS(2,2),TNS(3,3,2),TBNS(3,3,2),
        *          BSK(2,2),SSK(2,2),TNSK(3,3,2)
        *
0006    *     REAL*8 BS,SS,TNS,TBNS,BSK,SSK,TNSK
        * C
        * C     STAR TRACKER PARAMETERS
        * C           IN EACH CASE THE LAST SUBSCRIPT REFERS TO THE
        * C                 TRACKER USED
```

```
       * C                 BS       = BIAS - ACTUAL (RAD)
       * C                 SS       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
       * C                 TNS      = MISALIGNMENT ARRAY - TRANSFORMATION FROM
       * C                                   STAR TRACKER TO NOMINAL
       * C                 TBNS     = ORIENTATION ARRAY - TRANSFORMATION FROM
       * C                                   NOMINAL TO BODY
       * C                 BSK      = BIAS - KNOWLEDGE (RAD)
       * C                 SSK      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
       * C                 TNSK     = MISALIGNMENT KNOWLEDGE ARRAY - TRANSFORMATION
       * C                                   FROM STAR TRACKER TO NOMINAL
       * C
0007               INCLUDE 'MEASOUT.COM'
0008   *           COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
       *        .          RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
       *        .          EDHS(2),EDVS(2),EDHL,EDVL
0009   *           REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
       *        .          EDHS,EDVS,EDHL,EDVL
       * C
       * C
       * C         MEASUREMENT OUTPUT PARAMETERS
       * C                 MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
       * C                                   (KM,KM/SEC)
       * C                 EMXG     = ESTIMATED POSITION/VELOCITY STATE
       * C                                   MEASUREMENT - HGPS
       * C                 RGPS     = STATE MEASUREMENT NOISE COVARIANCE
       * C                                   (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
       * C                 DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
       * C                                   FROM BORESIGHT - START (RAD)
       * C                 DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
       * C                                   FROM BORESIGHT - START (RAD)
       * C                 EDHS     = ESTIMATED STAR MEASUREMENT HORIZONTAL
       * C                                   DEVIATION FROM BORESIGHT (RAD)
       * C                 EDVS     = ESTIMATED STAR MEASUREMENT VERTICAL
       * C                                   DEVIATION FROM BORESIGHT (RAD)
       * C                 MS       = STAR MEASUREMENT UNIT VECTOR (SECOND
       * C                                   SUBSCRIPT REFERS TO TRACKER) - START
       * C                 RS       = STAR MEASUREMENT NOISE COVARIANCE
       * C                                   (KNOWLEDGE) - START (RAD**2)
       * C                 DHCL     = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
       * C                                   FROM BORESIGHT - LAMKT (RAD)
       * C                 DVCL     = LANDMARK MEASUREMENT VERTICAL DEVIATION
       * C                                   FROM BORESIGHT - LAMKT (RAD)
       * C                 EDHL     = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
       * C                                   DEVIATION FROM BORESIGHT (RAD)
       * C                 EDVL     = ESTIMATED LANDMARK MEASUREMENT VERTICAL
       * C                                   DEVIATION FROM BORESIGHT (RAD)
       * C                 LMU      = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
       * C                 RL       = LANDMARK MEASUREMENT NOISE COVARIANCE
       * C                                   (KNOWLEDGE) - LAMKT (RAD**2)
       * C
       * C
         C         DIMENSION
0010               DIMENSION U(3)
0011               REAL*8 U,GAUSS
0012               DATA IFLAG/0/
         C**************************************************************
         C         GENERATE MEASUREMENT UNIT VECTOR IN STAR TRACKER COORDINATES
         C**************************************************************
```

```
0013          U(1)=0.
0014          U(2)=0.
0015          U(3)=1.
      C**************************************************************
      C       ADD BIAS AND NOISE TO THE IDEAL MEASUREMENT AND
      C                SUBTRACT OFF BIAS KNOWLEDGE
      C**************************************************************
0016          DO 10 I=1,2
0017  10      MS(I,K)=U(I)+BS(I,K)+GAUSS(0.,SS(I,K))-BSK(I,K)
      C**************************************************************
      C       NORMALIZE THE VECTOR
      C**************************************************************
0018          MS(3,K)=SQRT(1.-MS(1,K)**2-MS(2,K)**2)
0019          DVCS(K)=ASIN(MS(1,K))
0020          DHCS(K)=ASIN(MS(2,K)/COS(DVCS(K)))
      C**************************************************************
      C       GENERATE STAR TRACKER NOISE COVARIANCE
      C**************************************************************
0021          DO 20 I=1,2
0022          DO 20 J=1,2
0023  20      RS(I,J,K)=0.
0024          DO 30 I=1,2
0025  30      RS(I,I,K)=SSK(I,K)**2
0026          RETURN
0027          END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	298	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	4	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	48	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 STARPAR	560	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 MEASOUT	648	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		START

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-000001E0	R*8	DHCL	5-000001E8	R*8	DVCL	5-00000278	R*8	EDHL	5-00000280	R*8	EDVL
2-0000001C	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-00000018	I*4	IFLAG
2-00000020	I*4	J	AP-0000004@	I*4	K						

ARRAYS

Address	Type	Name	Bytes	Dimensions
4-00000000	R*8	BS	32	(2, 2)
4-00000160	R*8	BSK	32	(2, 2)
5-00000150	R*8	DHCS	16	(2)
5-00000160	R*8	DVCS	16	(2)
5-00000258	R*8	EDHS	16	(2)
5-00000268	R*8	EDVS	16	(2)
5-00000228	R*8	EMXG	48	(6)
5-000001F0	R*8	LMU	24	(3)
5-00000170	R*8	MS	48	(3, 2)
5-00000000	R*8	MX	48	(6)
5-00000030	R*8	RGPS	288	(6, 6)
5-00000208	R*8	RL	32	(2, 2)
5-000001A0	R*8	RS	64	(2, 2, 2)
4-00000020	R*8	SS	32	(2, 2)
4-00000180	R*8	SSK	32	(2, 2)
4-000000D0	R*8	TBNS	144	(3, 3, 2)
4-00000040	R*8	TNS	144	(3, 3, 2)
4-000001A0	R*8	TNSK	144	(3, 3, 2)
2-00000000	R*8	U	24	(3)

LABELS

Address	Label	Address	Label	Address	Label
**	10	**	20	**	30

FUNCTIONS AND SUBROUTINES REFERENCED

GAUSS MTH$DASIN MTH$DCOS MTH$DSQRT

Total Space Allocated = 1566 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          1.55 seconds
Elapsed Time:      15.94 seconds
Page Faults:       355
Dynamic Memory:    160 pages
```

2.6.3 Landmark Tracker Model (LAMKT)

Subroutine LAMKT simulates a measurement made by the landmark tracker. The landmark position vector is established in earth fixed coordinates through knowledge of the landmark's longitude, latitude, and altitude, passed through common. This position vector is then transformed to inertial coordinates. A line-of-sight vector to the landmark from the spacecraft is established by subtracting from the landmark position vector the spacecraft position vector. The line-of-sight vector is transformed to exact landmark tracker coordinates where it is normalized to a unit vector. The exact DH and DV measurements are computed based on the computed line-of-sight unit vector. These DH and DV measurements are corrupted with bias and noise and compensated by subtraction of bias knowledge. A unit vector is reestablished from the corrupt measurement and the noise covariance is established. All data output is through common.

Processing Requirements

The landmark location on the surface of the earth in Local Landmark Coordinates (Figure VII) will be a function of the altitude (A_L) above the earth's mean radius.

$$\underline{L}_L = \begin{bmatrix} A_L \\ 0 \\ 0 \end{bmatrix}$$

However, in earth fixed corrdinates, the landmark will have the earth's mean radius (r_E) added to the altitude. Using the angular transformation from local landmark to earth fixed coordinates produces

$$\underline{L}_E = {}_E T_L \, \underline{L}_L = {}_E T_L \begin{bmatrix} r_E + A_L \\ 0 \\ 0 \end{bmatrix} = (r_E + A_L) \begin{bmatrix} CLC\lambda & -SL & -CLS\lambda \\ SLC\lambda & CL & -SLS\lambda \\ S\lambda & 0 & C\lambda \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$= (r_E + A_L) \begin{bmatrix} CLC\lambda \\ SLC\lambda \\ S\lambda \end{bmatrix}$$

As shown in Figure A-32 the position vector of the spacecraft ($P_{S/C}$), when subtracted from the landmark position in some coordinate frame, will provide the measurement vector ($\underline{M}$).

$$\underline{M}_I = ({}_I T_E \underline{L}_E) - \underline{P}_{S/CI}$$

Accounting for hardware misalignments, the same measurement vector in landmark tracker coordinates is:

$$\underline{M}_\ell = {}_\ell T_I \, (({}_I T_E \underline{L}_E) - \underline{P}_{S/CI})$$

$$= {}_\ell T_E \underline{L}_E - T_I \underline{P}_{S/CI}$$

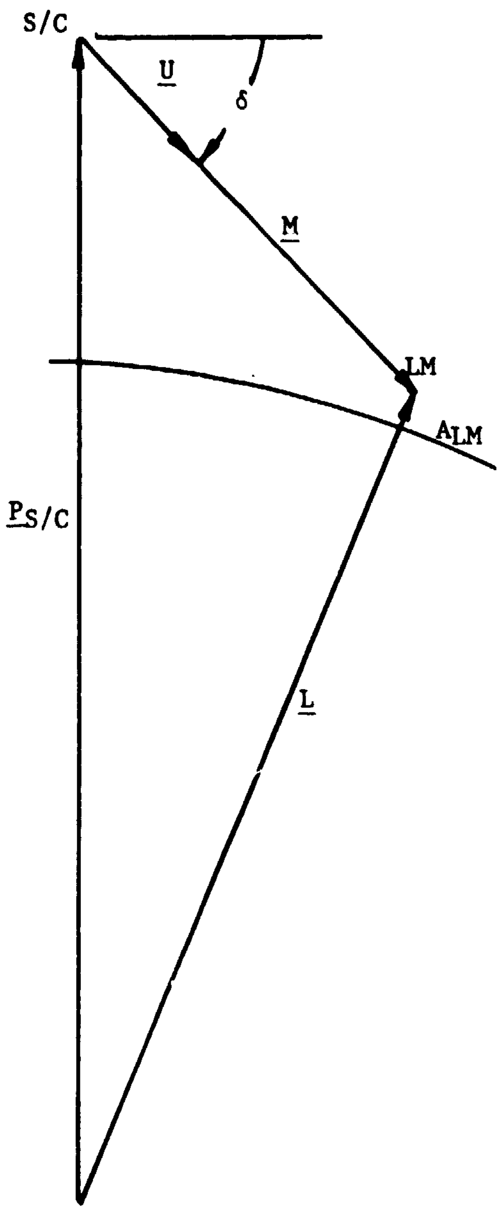


Figure A-32

Landmark Tracker Geometry - Sighting Plane

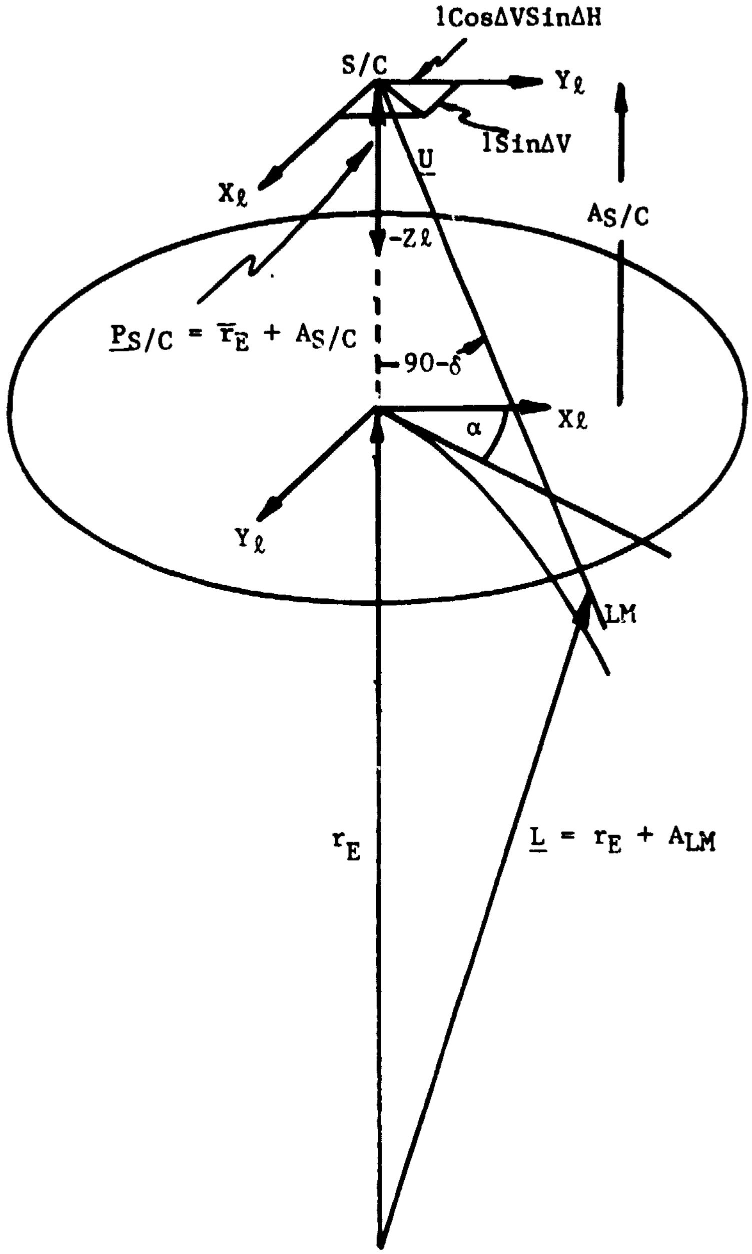


Figure A-33

Landmark Tracker Geometry - General

From examination of Figure A-33, the unit measurement vector in landmark tracker coordinates is:

$$\underline{U\ell} = \frac{\underline{M\ell}}{|\underline{M\ell}|} = \begin{bmatrix} U\ell x \\ U\ell y \\ U\ell z \end{bmatrix} = \begin{bmatrix} \mathrm{Sin}\Delta V \\ \mathrm{Cos}\Delta V \mathrm{Sin}\Delta H \\ -\mathrm{Cos}\Delta V \mathrm{Cos}\Delta H \end{bmatrix}$$

However, the tracker instrument has no sensitivity to projections along its boresight axis. Therefore, the tracker response to the unit vector $\underline{U\ell}$ will be:

$$\underline{U}'\ell = \begin{bmatrix} U\ell x \\ U\ell y \end{bmatrix} = \begin{bmatrix} \mathrm{Sin}\Delta V \\ \mathrm{Cos}\Delta V \mathrm{Sin}\Delta H \end{bmatrix}$$

producing aΔH and ΔV as shown in Figure A-33 as sensor outputs. Since the sensor output will be corrupted by bias and noise, the sensed measurement will be:

$$\underline{Z\ell} = \begin{bmatrix} \Delta Hs \\ \Delta Vs \end{bmatrix} = \begin{bmatrix} \Delta H + b_H + V_H \\ \Delta V + b_V + V_V \end{bmatrix}$$

Where:

b_H, b_V = Component landmark tracker bias.
V_H, V_V = Component landmark tracker zero mean random noise, N $(0, \sigma^2)$.

The component biases and standard deviations (σ) are user selectables.

The landmark tracker measurement may be compensated for knowledge of instrument bias. The bias knowledge may be a priori or through estimation. The compensated sensor output will be:

$$\hat{\underline{Z}}_\ell = \begin{bmatrix} \Delta Hc \\ \Delta Vc \end{bmatrix} = \begin{bmatrix} \Delta Hs - \hat{b}_H \\ \Delta Vs - \hat{b}_V \end{bmatrix}$$

Where:
LM = the landmark being used = f(L, λ, A_L)

A_L = the altitude of the landmark above the mean radius of the earth

L = longitude of the landmark

λ = latitude of the landmark

$\underline{L}$ = vector position of the landmark relative to the center of the earth

$\underline{P}_{s/c}$ = vector position of the spacecraft relative to the center of the earth

$A_{s/c}$ = altitude of the spacecraft above the mean radius of the earth

$\underline{M}$ = measurement vector from the spacecraft to the landmark

$\underline{U}$ = unit vector along $\underline{M}$

H = the landmark tracker horizontal plane angular deflection from the boresight axis

V = the landmark tracker vertical plane angular deflection from the boresight axis

Landmark Tracker Model
(LAMKT)

Generate Landmark Position in Earth Fixed Coordinates
Bring Landmark Position into Inertial Space
Subtract Spacecraft Position Vector
Transform To Landmark Tracker Coordinates
Normalize Line-of-Sight Vector
Generate DN and DV Measurements
Corrupt Measurement With Bias and Noise
Compensate for Bias Knowledge
Construct Measurement Unit Vector
Construct Measurement Noise Covariance

Figure A-34

```
0001              SUBROUTINE LAMKT
          C*************************************************************
          C
          C       SUBROUTINE LAMKT SIMULATES A MEASUREMENT MADE BY THE
          C       LANDMARK TRACKER. IT DOES SO BY SUBTRACTING FROM PERFECT
          C       KNOWLEDGE OF THE LANDMARK POSITION (LONGITUDE, LATITUDE,
          C       AND ALTITUDE) PERFECT KNOWLEDGE OF THE SPACECRAFT POSITION.
          C       TO THIS VECTOR BIAS AND NOISE ARE ADDED WHILE IN LANDMARK
          C       TRACKER COORDINATES. COMPENSATION FOR KNOWN BIAS IS MADE
          C       BY SUBTRACTING THIS KNOWLEDGE AT THIS POINT. COMPENSATION
          C       FOR KNOWN LANDMARK TRACKER MISALIGNMENT IS MADE WHEN THE
          C       RESULTING VECTOR IS TRANSFORMED BACK TO INERTIAL SPACE.
          C
          C       INPUT VARIABLES
          C               RE      = AVERAGE RADIUS OF THE EARTH (KM)
          C               AL      = LANDMARK ALTITUDE (KM)
                          LON     = LANDMARK LONGITUDE (DEG)
          C               LAT     = LANDMARK LATITUDE (DEG)
          C               T       = TIME (TBD)
          C               P       = SPASECRAFT POSITION VECTOR (3*1) (KM)
          C               Q       = EXACT INERTIAL TO BODY QUATERNIAN (4*1)
          C               TBNL    = LANDMARK TRACKER ORIENTATION ARRAY
          C                           (NOMINAL TO BODY) (3*3)
          C               TNL     = LANDMARK TRACKER MISALIGNMENT ARRAY
          C                           (TRACKER TO NOMINAL) (3*3)
          C               BL      = LANDMARK TRACKER BIAS (2*1) (RAD)
          C               SL      = LANDMARK TRACKER NOISE STANDARD DEVIATION
          C                           (2*1) (RAD)
          C               BKL     = LANDMARK TRACKER BIAS KNOWLEDGE (2*1)
          C                           (RAD)
          C               SKL     = LANDMARK TRACKER NOISE STANDARD
          C                           DEVIATION (2*1) (RAD)
          C               TNLK    = LANDMARK TRACKER MISALIGNMENT KNOWLEDGE
          C                           ARRAY (TRACKER TO NOMINAL) (3,3)
          C               SIGGCP  = POSITION UNCERTAINTY DUE TO CLOUDS
          C
          C       OUTPUT VARIABLES
          C               DHCL    = LANDMARK HORIZONTAL DEFLECTION (RAD)
          C               DVCL    = LANDMARK VERTICAL DEDLECTION (RAD)
          C               LMU     = SIMULATED LANDMARK TRACKER UNIT VECTOR
          C                           TO LANDMARK (3*1)
          C               RL      = LANDMARK TRACKER MEASUREMENT COVARIANCE
          C                           (3*3) (RAD)
          C
          C
          C       PROGRAMMED BY JACK MYERS 11JUNE1980
          C                       EXT 4443
          C
          C*************************************************************
          C       COMMON BLOCKS
0002              INCLUDE 'ENVIR.COM'
0003      *       COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
0004      *       REAL*8 STATE,PROFILE
          * C
          * C             REAL WORLD STATE PARAMETERS
          * C
          * C                   STATE     STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
```

```
              * C                  PROFILE    ATTITUDE PROFILE-TIME (SEC) VS
              * C                             INERTIAL ANGULAR RATES (RAD/SEC)
              * C                  INIT       INTEGRATION INITIALIZATION KEY (-1)
              * C
      0005             INCLUDE 'TIME.COM'
              * C
      0006    *       COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
              *      .             ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
      0007    *       REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
              *      .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
              * C
              * C             THESE ARE THE TIME REFERENCE FRAMES
              * C
              * C                  TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
              * C                  TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
              * C                  TSTOP      RUN TERMINATION TIME (SEC)
              * C                  TIA        ATTITUDE INTEGRATION TIME (SEC)
              * C                  DEL           "          "      STEP SIZE (SEC)
              * C                  TIN        POSITION INTEGRATION TIME (SEC)
              * C                  DTN           "          "      STEP SIZE (SEC)
              * C                  DATEO      DATE OF FLIGHT EPOCH (JD)
              * C                  DATER      DATE OF 1950 EPOCH (JD)
              * C                  TZERO      START TIME IN SECS. SINCE DATEO
              * C                  TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
              * C                  TIS        REAL WORLD REFERENCE TIME (SEC)
              * C                  TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
              * C                  DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
              * C                             TOO LARGE AT MEASUREMENT TIME
              * C                  TPRINT     TIME FOR PRINT (SEC)
              * C                  DTPRINT    INCREMENT ON TPRINT (SEC)
              * C
      0008             INCLUDE 'DEBUG.COM'
      0009    *       COMMON /DEBUG/ IENTER,IDEBUG
              * C
              * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
              * C
              * C                  IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
              * C                  IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
              * C
      0010             INCLUDE 'LMTPAR.COM'
      0011    *        COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
              *                BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
      0012    *        REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
              *                THET
              * C
              * C
              * C      LANDMARK TRACKER PARAMETERS
              * C              AL      = ALTITUDE OF LANDMARK (KM)
              * C              LON     = LONGITUDE OF LANDMARK (DEG)
              * C              LAT     = LATITUDE OF LANDMARK (DEG)
              * C              TBNL    = ORIENTATION ARRAY FOR LANDMARK TRACKER
              * C                               NOMINAL TO BODY
              * C              TNL     = MISALIGNMENT ARRAY - ACTUAL
              * C                               TRACKER TO NOMINAL
              * C              BL      = BIAS - ACTUAL (RAD)
              * C              SL      = NOISE STANDARD DEVIATION - ACTUAL (RAD)
              * C              BKL     = BIAS - KNOWLEDGE (RAD)
```

```
               * C            THET     = LOOK ANGLE (RAD)
               * C            SKL      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
               * C            TIEO     = INITIAL EARTH FIXED TO INERTIAL
               * C                             TRANSFORMATION
               * C            TNLK     = MISALIGNMENT ARRAY KNOWLEDGE
               * C                             TRACKER TO NOMINAL
               * C            SIGGCP   = POSITION UNCERTAINTY DUE TO CLOUDS
               * C
       0013              INCLUDE 'MEASOUT.COM'
       0014    *         COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
               *        .        RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
               *        .        EDHS(2),EDVS(2),EDHL,EDVL
       0015    *         REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
               *        .        EDHS,_DVS,EDHL,EDVL
               * C
               * C
               * C       MEASUREMENT OUTPUT PARAMETERS
               * C            MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
               * C                             (KM,KM/SEC)
               * C            EMXG     = ESTIMATED POSITION/VELOCITY STATE
               * C                             MEASUREMENT - HGPS
               * C            RGPS     = STATE MEASUREMENT NOISE COVARIANCE
               * C                             (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
               * C            DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
               * C                             FROM BORESIGHT - START (RAD)
               * C            DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
               * C                             FROM BORESIGHT - START (RAD)
               * C            EDHS     = ESTIMATED STAR MEASUREMENT HORIZONTAL
               * C                             DEVIATION FROM BORESIGHT (RAD)
               * C            EDVS     = ESTIMATED STAR MEASUREMENT VERTICAL
               * C                             DEVIATION FROM BORESIGHT (RAD)
               * C            MS       = STAR MEASUREMENT UNIT VECTOR (SECOND
               * C                             SUBSCRIPT REFERS TO TRACKER) - START
               * C            RS       = STAR MEASUREMENT NOISE COVARIANCE
               * C                             (KNOWLEDGE) - START (RAD**2)
               * C            DHCL     = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
               * C                             FROM BORESIGHT - LAMKT (RAD)
               * C            DVCL     = LANDMARK MEASUREMENT VERTICAL DEVIATION
               * C                             FROM BORESIGHT - LAMKT (RAD)
               * C            EDHL     = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
               * C                             DEVIATION FROM BORESIGHT (RAD)
               * C            EDVL     = ESTIMATED LANDMARK MEASUREMENT VERTICAL
               * C                             DEVIATION FROM BORESIGHT (RAD)
               * C            LMU      = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
               * C            RL       = LANDMARK MEASUREMENT NOISE COVARIANCE
               * C                             (KNOWLEDGE) - LAMKT (RAD**2)
               * C
       0016              INCLUDE 'NSTATE.COM'
               * C
       0017    *       COMMON /NSTATE/ XD(6),X(6),RADM,RADE
       0018    *       REAL*8 XD,X,RADM,RADE
               * C
               * C            POSITION STATE AND CONSIDERED PARAMETERS
               * C
               * C                 XD       STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
               * C                 X        STATE POSITION PARAMETERS (KM AND KM/SEC)
               * C                 RADM     RADIUS OF THE MOON (KM)
```

```
            * C                     RADE       EARTH DETECTABLE RADIUS (KM)
            * C
            * C
     0019          INCLUDE 'TARG.TS.COM'
     0020   *      COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
     0021   *      LOGICAL JFLAG
     0022   *      REAL*8 PI,TPI
            * C
            * C               MEASUREMENT SPECIFICATIONS
            * C
            * C                     MTYPE      MEASUREMENT TYPE
            * C                     JFLAG      SET FOR STAR OBSTRUCTION
            * C                     MCODE       "   "  MEASUREMENT PROCESSING
            * C                     PI         PI
            * C                     TPI        2*PI
            * C
     0023          EQUIVALENCE (Q,STATE(7)),(P,STATE),(DELT,DELTL),(B,BL),(S,SL),
            .          (BK,BKL),(SK,SKL),(T,TIME),(RE,RADE)
     0024          DIMENSION LE(3),LI(3),MI(3),TBL(3,3),TLB(3,3),TBI(3,3),TLI(3,3),
            .      ML(3),UML(3),P(3),TIE(3,3)
     0025          REAL*8 DELT,DELTL,DH,DV,RTD,TO,TIE,LE,LI,MI,ML,
            .          TBI,TBL,TLB,TLI,UML,B,BK,Q,RE,S,T,P,GAUSS
     0026          DATA IFLAG/0/
              C************************************************************
              C        GENERATE THE LANDMARK POSITION IN EARTH FIXED COORDINATES
              C************************************************************
     0027          RTD=180./PI       ! RADIAN TO DEGREE CONVERSION
     0028          LE(1)=(RE+AL)*COS(LON/RTD)*COS(LAT/RTD)
     0029          LE(2)=(RE+AL)*SIN(LON/RTD)*COS(LAT/RTD)
     0030          LE(3)=(RE+AL)*SIN(LAT/RTD)
              C************************************************************
              C        BRING LANDMARK POSITION INTO INERTIAL SPACE
              C************************************************************
     0031          CALL WET(TIEO,TIE)           ! GENERATE TIE
     0032          CALL MATAB(TIE,LE,LI,3,3,1)
              C************************************************************
              C        SUBTRACT SPACECRAFT POSITION IN INERTIAL SPACE
              C************************************************************
     0033          DO 10 I=1,3
     0034     10   MI(I)=LI(I)-P(I)
              C************************************************************
              C        TRANSFORM TO LANDMARK TRACKER COORDINATES
              C************************************************************
     0035          CALL AMAT(Q,TBI)
     0036          CALL MATAB(TBNL,TNL,TBL,3,3,3)
     0037          CALL MINV3(TBL,TLB)
     0038          CALL MATAB(TLB,TBI,TLI,3,3,3)
     0039          CALL MATAB(TLI,MI,ML,3,3,1)
              C************************************************************
              C        NORMALIZE THE MEASUREMENT VECTOR
              C************************************************************
     0040          CALL UNIT(ML,UML,3)
              C************************************************************
              C        GENERATE DH AND DV MEASUREMENTS FROM UNIT MEASUREMENT
              C                VECTOR
              C************************************************************
     0041          DV=ASIN(UML(1))
```

```
0042          DH=ASIN(UML(2)/COS(DV))
      C*************************************************************
      C       CORRUPT THE M ASUREMENT VECTOR WITH BIAS AND NOISE AND
      C               COMPENSATE FOR THE BIAS KNOWLEDGE
      C*************************************************************
0043          SL(1)=.001*SIGGCP/VMAG(MI,3)   !CHANGE DISTANCE TO ANGLE IN RAD.
0044          SL(2)=SL(1)                    !DITTO
0045          DHCL=DH+BL(1)+GAUSS(0.,SL(1))-BKL(1)
0046          DVCL=DV+BL(2)+GAUSS(0.,SL(2))-BKL(2)
      C*************************************************************
      C       RECONSTRUCT UNIT VECTOR FROM COMPENSATED MEASUREMENTS
      C               POSSIBLY NOT NEEDED AT THIS POINT
      C*************************************************************
0047          LMU(1)=COS(DVCL)*SIN(DHCL)
0048          LMU(2)=SIN(DVCL)
0049          LMU(3)=COS(DVCL)*COS(DHCL)
      C*****************************************************************
      C       FORM MEASUREM_NT NOISE COVARIANCE
      C*************************************************************
0050          DO 20 I=1,2
0051          DO 20 J=1,2
0052    20    RL(I,J)=0.
0053          DO 30 I=1,2
0054    30    RL(I,I)=SL(I)*SL(I)
0055          RETURN
0056          END
```


PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	501	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	12	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	748	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	LMTPAR	392	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	MEASOUT	648	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
9	TARGETS	36	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		LAMKT

VARIABLES

Address	Type	Name
6-00000000	R*8	AL
4-00000070	R*8	DATER
2-000001E8	R*8	DH
4-00000080	R*8	DTPRINT
7-00000280	R*8	EDVL
2-00000208	I*4	IFLAG
9-0000000C	L*4	JFLAG
9-00000000	I*4	MTYPE
8-00000068	R*8	RADE
6-00000088	R*8	S
2-00000200	R*8	TO
4-00000028	R*8	TIN
4-00000008	R*8	TNEXT
4-00000010	R*8	TSTOP

Address	Type	Name
6-000000A8	R*8	B
4-00000020	R*8	DEL
7-000001E0	R*8	DHCL
2-000001F0	R*8	DV
2-0000020C	I*4	I
3-00000190	I*4	INIT
6-00000010	R*8	LAT
9-00000008	I*4	NS
8-00000060	R*8	RADM
6-00000178	R*8	SIGGCP
6-00000180	R*8	THET
4-00000058	R*8	TIS
9-0000001C	R*8	TPI
4-00000040	R*8	TZERO

Address	Type	Name
6-000000C8	R*8	BK
2-00000000	R*8	DELT
4-00000068	R*8	DTA
7-000001E8	R*8	DVCL
5-00000004	I*4	IDEBUG
9-00000004	I*4	IS
6-00000008	R*8	LON
9-00000014	R*8	PI
8-00000068	R*8	RE
6-000000D8	R*4	SK
4-00000018	R*8	TIA
4-00000060	R*8	TISN
4-00000078	R*8	TPRINT

Address	Type	Name
4-00000038	R*8	DATEO
2-00000000	R*8	DELTL
4-00000030	R*8	DTN
7-00000278	R*8	EDHL
5-00000000	I*4	IENTER
2-00000210	I*4	J
9-00000010	I*4	MCODE
3-00000030	R*8	Q
2-000001F8	R*8	RTD
4-00000000	R*8	T
4-00000000	R*8	TIME
4-00000048	R*8	TMEAS
4-00000050	R*8	TRACK

ARRAYS

Address	Type	Name	Bytes	Dimensions
6-000000C8	R*8	BKL	16	(2)
6-000000A8	R*8	BL	16	(2)
7-00000150	R*8	DHCS	16	(2)
7-00000160	R*8	DVCS	16	(2)
7-00000258	R*8	EDHS	16	(2)
7-00000268	R*8	EDVS	16	(2)
7-00000228	R*8	EMXG	48	(6)
2-00000008	R*8	LE	24	(3)
2-00000020	R*8	LI	24	(3)
7-000001F0	R*8	LMU	24	(3)
2-00000038	R*8	MI	24	(3)
2-00000170	R*8	ML	24	(3)
7-00000170	R*8	MS	48	(3, 2)
7-00000000	R*8	MX	48	(6)
3-00000000	R*8	P	24	(3)
3-00000050	R*8	PROFILE	320	(10, 4)
7-00000030	R*8	RGPS	288	(6, 6)
7-00000208	R*8	RL	32	(2, 2)
7-000001A0	R*8	RS	64	(2, 2, 2)
6-000000D8	R*8	SKL	16	(2)
6-00000088	R*8	SL	16	(2)
3-00000000	R*8	STATE	80	(10)
2-000000E0	R*8	TBI	72	(3, 3)
2-00000050	R*8	TBL	72	(3, 3)
6-00000018	R*8	TBNL	72	(3, 3)
2-000001A0	R*8	TIE	72	(3, 3)
6-00000130	R*8	TIEO	72	(3, 3)
2-00000098	R*8	TLB	72	(3, 3)
2-00000128	R*8	TLI	72	(3, 3)
6-00000060	R*8	TNL	72	(3, 3)
6-000000E8	R*8	TNLK	72	(3, 3)
2-00000188	R*8	UML	24	(3)
8-00000030	R*8	X	48	(6)

```
  8-00000000  R*8  XD                    48  (6)
```

LABELS

Address	Label	Address	Label	Address	Label
**	10	**	20	**	30

FUNCTIONS AND SUBROUTINES REFERENCED

```
AMAT        GAUSS       MATAB       MINV3       MTH$DASIN   MTH$DCOS    MTH$DSIN    UNIT
VMAG        WET
```

Total Space Allocated = 2997 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           2.60 seconds
  Elapsed Time:       35.98 seconds
  Page Faults:        400
  Dynamic Memory:     160 pages
```

2.7 Perform State Estimation (EST)

EST incorporates the latest GPS, star tracker, and GCP measurements into an estimate of the spacecraft state using an extended Carlson Square Root filter. The extended filter incorporates propagation of the state, the state transition matrix, and the process noise through integration of non linear differential equations. The square root filter has been selected to overcome numerical difficulties associated with small spacecraft computers.

The process, illustrated in Figure A-35 begins by constructing a single composit state out of navigation and attitude states. The two portions of the state were separated because they are propagated in dramatically different ways. The navigator state, X, is propagated through non-linear integration of the equations of motion while the attitude stae, E, is propagated by the gyros. Similarly the composite state transition and covariance matrices are constructed from the attitude and position components. The last composite array to be found is the process noise. However, the attitude portion of the process noise array is loaded with zeros because it was previously considered in gyro processing. After the composite process noise array has been loaded, the navigation process noise array is reinitialized to zero for the next measurement interval.

Following construction of the composite arrays, the covariance matrix is propagated forward to current time by the subroutine PROP. PROP uses the state transition matrix PHI to perform this function. PROP also reinitializes the state transition matrix to the identity matrix, and EST loads the independent attitude and navigation state transition matrices.

The measurement partials and the estimated measurement are computed by MPART. MPART uses the measurement code to determine which partials to generate. These partials are used along with the measurement residual, to estimate the state of the spacecraft. This process is performed in a systematic manner. First, the measurement residual is computed by subtracting the estimated measurement vector from the true measurement vectors. Each component of the measurement residual is used sequentially in a two step process to update the state and covariance and to reload the old covariance. It is necessary to reload the old covariance between each update because the original old covariance is no longer valid after one update.

Following state update, the navigation and attitude states are loaded with the new state estimate. The attitude quaternions are unitized to insure orthogonality, and the various transformation matrices are computed.

The final procedure in EST is to insure that the attitude covariance does not drop below some minimum required to insure stability. If the covariance were allowed to converge to zero, new measurements would be ignored and the state estimate would diverge.

STATE ESTIMATION MODULE (EST)

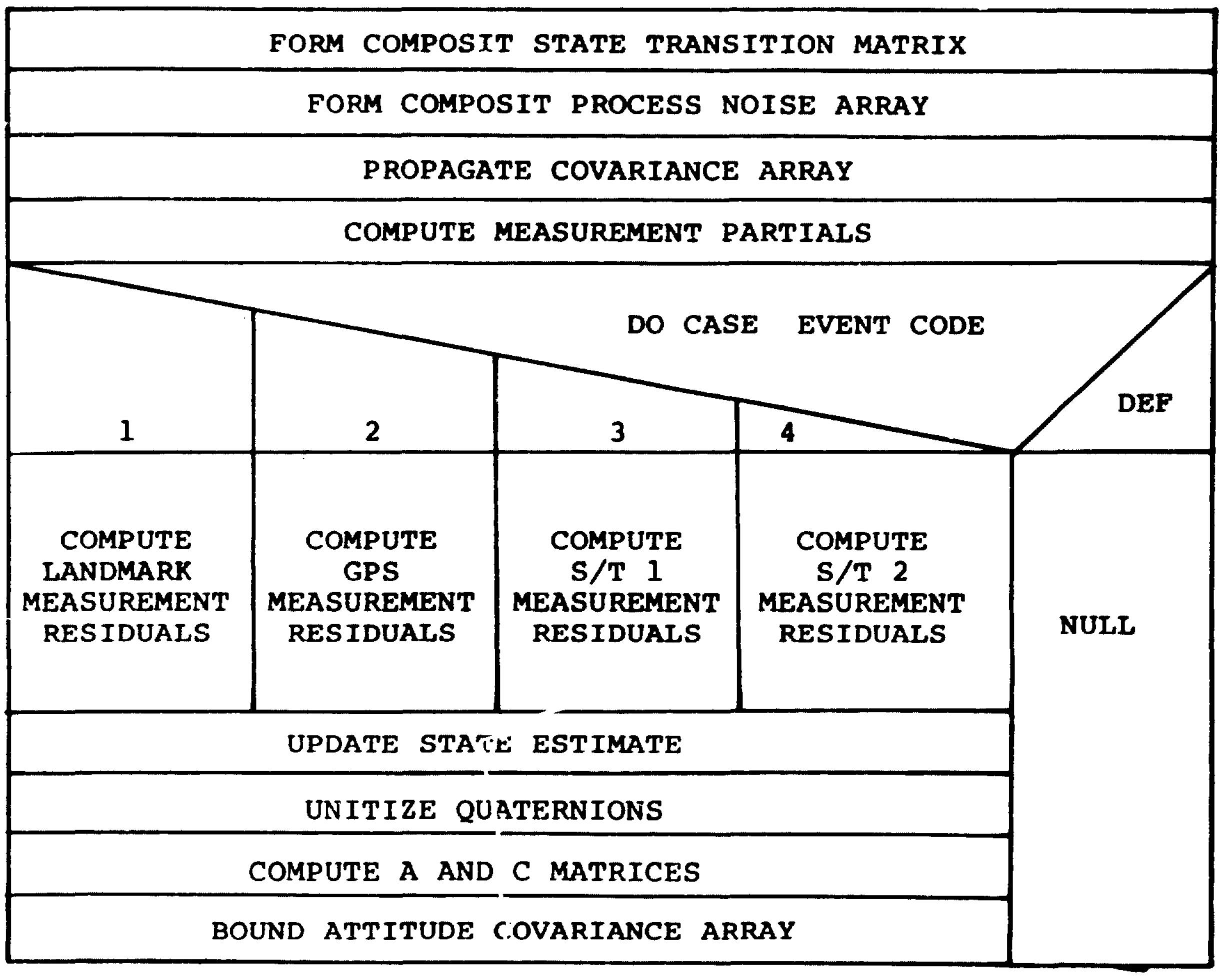


Figure A-35

```
0001          SUBROUTINE EST
       C*********************************************************************
       C
       C      SUBROUTINE EST REPLACES SUBROUTINE ESTIMAT BY PERFORMING THE
       C      ESTIMATION PROCESS USING THE MEASUREMENTS MADE BY THE
       C      LANDMARK TRACKER, THE GPS, AND STAR TRACKERS 1 AND 2.
       C      THE ESTIMATION STATE VECTOR HAS BEEN AUGMENTED TO 10 WHILE
       C      THERE ARE 12 CONSIDER PARAMETERS FOR A TOTAL COMBINED STATE
       C      OF 22.
       C
       C      INPUT PARAMETERS
       C              PN       = STATE TRANSITION MATRIX INITIAL CONDITIONS
       C                               FOR THIS INTERVAL
       C              PHIN     = NAVIGATION STATE TRANSITION MATRIX INTEGRATED
       C                               TO THIS TIME POINT
       C              PHIA     = ATTITUDE STATE TRANSITION MATRIX INTEGRATED
       C                               TO THIS TIME POINT
       C              Q        = NAVIGATION PROCESS NOISE
       C              MCODE    = FLAG DETERMINING THE MEASUREMENTS TO BE USED
       C              DHCL     = HORIZONTAL LANDMARK MEASUREMENT
       C              DVCL     = VERTICAL LANDMARK MEASUREMENT
       C              PDHL     = PARTIAL OF HORIZONTAL LANDMARK MEASUREMENT
       C              PDVL     =    "     "   VERTICAL       "         "
       C              SIGGCP   = STANDARD DEVIATION OF LMT POSITION
       C                               MEASUREMENT
       C
       C              MX       = GPS MEASUREMENT VECTOR
       C              PON      = OLD NAVIGATION COVARIANCE
       C              POA      = OLD ATTITUDE COVARIANCE
       C              ETC
       C
       C*********************************************************************
0002          REAL*8 VDOT,PDHDX,PDVDX
       C      COMMON BLOCKS
0003          INCLUDE 'DEBUG.COM'
0004  *     COMMON /DEBUG/ IENTER,IDEBUG
      * C
      * C            USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
      * C
      * C                 IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
      * C                 IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
      * C
0005          INCLUDE 'TARGETS.COM'
0006  *     COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
0007  *     LOGICAL JFLAG
0008  *     REAL*8 PI,TPI
      * C
      * C            MEASUREMENT SPECIFICATIONS
      * C
      * C                 MTYPE      MEASUREMENT TYPE
      * C                 JFLAG      SET FOR STAR OBSTRUCTION
      * C                 MCODE       "   "  MEASUREMENT PROCESSING
      * C                 PI         PI
      * C                 TPI        2*PI
      * C
0009          INCLUDE 'PHIA.COM'
0010  *     COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
```

```
00200          *           COVA(16,16),POA(16,16),QMAX
00300   0011   *       REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
00400          * C
00500          * C              THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
00600          * C
00700          * C                   PA         ATTITUDE STATE TRANSITION MATRIX
00800          * C                   TA            "     PARAMETER TRANSITION MATRIX
00900          * C                   PDA           "     DERIVATIVE OF TRANSITION MATRICES
01000          * C                   PHIA          "     AGGREGATE TRANSITION MATRIX
01100          * C                   COVA          "     NEW COVARIANCE MATRIX
01200          * C                   POA           "     PREVIOUS COVARIANCE MATRIX
01300          * C                   QMAX          "     COVARIANCE NORM MAX
01400          * C
03800   0012          INCLUDE 'GAIN.COM'
03900          * C
03900          * C
03900   0013   *      COMMON /GAIN/GAIN(22)
03900   0014   *      REAL*8 GAIN
03900          * C
03900          * C
03900   0015          INCLUDE 'RESIDUALS.COM'
04000          * C****************************************************************
04000          * C                                                               *
04000          * C--- RESIDUALS.COM CONTAINS THE RESIDUALS VALUES FOR THE GPS,    *
04000          * C--- LANDMARK TRACKER, AND STAR TRACKER MEASUREMENTS             *
04000          * C                                                               *
04000          * C****************************************************************
04000          * C
04000   0016   *      COMMON /RESIDUALS/DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,
04000          *     *                    DXMGPS(6)
04000   0017   *      REAL*8 DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,DXMGPS
04000          * C
04000   0018          INCLUDE 'PHIN.COM'
04100   0019   *     COMMON /PHIN/ PN(6,6),PDN(6,6),PHIN(6,6),COVN(6,6),
04100          *    *         PON(6,6)
04100   0020   *     REAL*8 PN,PDN,PHIN,COVN,PON
04100          * C
04100          * C              THESE ARE THE NAVIGATION TRANSITION AND COVARIANCE ARRAYS
04100          * C
04100          * C                   PN         POSITION STATE TRANSITION MATRIX
04100          * C                   PDN           "     DERIVATIVE OF TRANSITION MATRIX
04100          * C                   PHIN          "     AGGREGATE TRANSITION MATRIX
04100          * C                   COVN          "     NEW COVARIANCE MATRIX
04100          * C                   PON           "     PREVIOUS COVARIANCE MATRIX
04100          * C
04100   0021          INCLUDE 'UPDT.COM'
04200          * C
04200   0022   *     COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
04200   0023   *     REAL*8 QN,QA,Q,QDOT
04200          * C
04200          * C              STATE ESTIMATION PARAMETERS
04200          * C
04200          * C                   QN         NAV. DYN. NOISE COVARIANCE DIAGONAL
04200          * C                   QA         MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
04200          * C                   Q          CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
04200          * C                   QDOT       DIFFERENTIAL OF Q
04200          * C
```

```
04200   0024         INCLUDE 'ASTATE.COM'
04300          * C
04300   0025   *      COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
04300   0026   *      REAL*8 DE,E,WD,SF,D,DD
04300          * C
04300          * C              ATTITUDE STATE AND CONSIDERED PARAMETERS
04300          * C
04300          * C                    D.        DIFERENTIAL OF QUATERNIONS
04300          * C                    E         QUATERNIONS
04300          * C                    WD        GYRO DRIFT RATE (RAD/SEC)
04300          * C                    SF        GYRO SCALE FACTOR
04300          * C                    D         GYRO NON-ORTHOGANALITY (RAD)
04300          * C                    DO        GYRO RELATIVE ORIENTATION (RAD)
04300          * C
04300   0027         INCLUDE 'NSTATE.COM'
04400          * C
04400   0028   *      COMMON /NSTATE/ XD(6),X(6),RADM,RADE
04400   0029   *      REAL*8 XD,X,RADM,RADE
04400          * C
04400          * C              POSITION STATE AND CONSIDERED PARAMETERS
04400          * C
04400          * C                    XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
04400          * C                    X         STATE POSITION PARAMETERS (KM AND KM/SEC)
04400          * C                    RADM      RADIUS OF THE MOON (KM)
04400          * C                    RADE      EARTH DETECTABLE RADIUS (KM)
04400          * C
04400          * C
04400   0030         INCLUDE 'TIME.COM'
00100          * C
00200   0031   *      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
00300          *     .             ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
00400   0032   *      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
00500          *     .      TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
00600          * C
00700          * C              THESE ARE THE TIME REFERENCE FRAMES
00800          * C
00900          * C                    TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
01000          * C                    TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
01100          * C                    TSTOP     RUN TERMINATION TIME (SEC)
01200          * C                    TIA       ATTITUDE INTEGRATION TIME (SEC)
01300          * C                    D L          "          "     STEP SIZE (SEC)
01400          * C                    TIN       POSITION INTEGRATION TIME (SEC)
01500          * C                    DTN          "          "     STEP SIZE (SEC)
01600          * C                    DATEO     DATE OF FLIGHT EPOCH (JD)
01700          * C                    DATER     DATE OF 1950 EPOCH (JD)
01800          * C                    TZERO     START TIME IN SECS. SINCE DATEO
01900          * C                    TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
02000          * C                    TIS       REAL WORLD REFERENCE TIME (SEC)
02100          * C                    TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
02200          * C                    DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
02300          * C                              TOO LARGE AT MEASUREMENT TIME
02400          * C                    TPRINT    TIME OF PRINT (SEC)
02500          * C                    DTPRINT   INCREMENT ON TPRINT (SEC)
02600          * C
04500   0033         INCLUDE 'TMAT.COM'
04600   0034   *      COMMON /TMAT/ A(3,3),B(3,3),C(3,3),EM(4,3)
04600   0035   *      REAL*8 A,B,C,EM
```

```
04600          • C
04600          • C          TRANSFORMATION MATRICES
04600          • C
04600          • C               A          INERTIAL TO BODY AXES
04600          • C               B          GYRO TO BODY AXES
04600          • C               C          GYRO NON-ORTHOGONAL TO GYRO AXES
04600          • C               EM         BODY TO QUATERNIAN AXES
04600          • C
04600   0036            INCLUDE 'FILT.R.COM'
04700   0037   •      COMMON/FILTER/ IPN(6),IPA(16)     !IPN WAS IPN(11)    JACK
04700          • C
04700          • C          FILTER DATA CONSTANTS
04700          • C
04700          • C               IPN        ARRAY INDEX OF ESTIMATED POS PARAMETERS
04700          • C               IPA          "     "          "      ATT PARAMETERS
04700          • C
04700   0038            INCLUDE 'COMPOSIT.COM'
04800   0039   •        COMMON /COMPOSIT/ PHI(22,22),QT(22,22),COV(22,22),PO(22,22),
04800          •      .      IP(22),XT(22),P(22,22)
04800   0040   •        REAL*8 PHI,QT,COV,PO,XT,P
04800          • C
04800          • C             PHI       = COMPOSIT STATE TRANSITION MATRIX
04800          • C             QT        =      "      PROCESS NOISE ARRAY
04800          • C             COV       = NEWEST COVARIANCE ARRAY
04800          • C             PO        = OLD COVARIANCE ARRAY
04800          • C             IP        = ARRAY OF FLAGS INDICATING ESTIMATED AND
04800          • C                              CONSIDERED PARAMETERS
04800          • C             XT        = COMPOSIT ESTIMATED PLUS CONSIDERED
04800          • C                              STATE VECTOR
04800          • C             P         = INITIALIZED TRANSITION MATRIX FOR NEXT
04800          • C                              INTREVAL
04800          • C
04800   0041            INCLUDE 'ARRAYS.COM'
04900          • C
04900   0042   •      COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
04900          •      .          ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
04900   0043   •      REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
04900          • C
04900          • C          THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
04900          • C
04900          • C               T1 - T4     SINGLE DIMENSION ARRAYS
04900          • C               T11 - T77   DUAL DIMENSIONED ARRAYS
04900          • C               T11         DUAL ARRAY: OFF DIAGIONAL SET TO ZERO
04900          • C
04900   0044            INCLUDE 'PART.COM'
05000   0045   •        COMMON /PART/ PX(22),PY(22),PZ(22),PXD(22),PYD(22),PZD(22),
05000          •      .      PDHS(22,2),PDVS(22,2),PDHL(22),PDVL(22)
05000   0046   •        REAL*8 PX,PY,PZ,PXD,PYD,PZD,PDHS,PDVS,PDHL,PDVL
05000          • C
05000          • C      PARTIALS OF THE RESPECTIVE MEASUREMENTS MADE
05000          • C
05000          • C      FOR GPS
05000          • C             PX        = PARTIALS OF X POSITION MEASUREMENT
05000          • C             PY        =    "      "  Y    "          "
05000          • C             PZ        =    "      "  Z    "          "
05000          • C             PXD       =    "      "  X VELOCITY      "
05000          • C             PYD       =    "      "  Y    "          "
```

```
          * C            PZD      =   "     "   Z      "       "
          * C      FOR STAR TRACKER K (K IS THE SECOND PARAMETER)
          * C            PDHS     = PARTIALS OF HORIZONTAL DIFLECTION
          * C            PDVS     =   "     "   VERTICAL      "
          * C      FOR LANDMARK TRACKER
          * C            PDHL     = PARTIALS OF HORIZONTAL DIFLECTION
          * C            PDVL     =   "     "   VERTICAL      "
          * C
    0047               INCLUDE 'MEASOUT.COM'
    0048  *            COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
          *       .          RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
          *       .          EDHS(2),EDVS(2),EDHL,EDVL
    0049  *            REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
          *       .          EDHS, DVS,EDHL,EDVL
          * C
          * C
          * C      MEASUREMENT OUTPUT PARAMETERS
          * C            MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
          * C                           (KM,KM/SEC)
          * C            EMXG     = ESTIMATED POSITION/VELOCITY STATE
          * C                           MEASUREMENT - HGPS
          * C            RGPS     = STATE MEASUREMENT NOISE COVARIANCE
          * C                           (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
          * C            DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
          * C                           FROM BORESIGHT - START (RAD)
          * C            DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
          * C                           FROM BORESIGHT - START (RAD)
          * C            EDHS     = ESTIMATED STAR MEASUREMENT HORIZONTAL
          * C                           DEVIATION FROM BORESIGHT (RAD)
          * C            EDVS     = ESTIMATED STAR MEASUREMENT VERTICAL
          * C                           DEVIATION FROM BORESIGHT (RAD)
          * C            MS       = STAR MEASUREMENT UNIT VECTOR (SECOND
          * C                           SUBSCRIPT REFERS TO TRACKER) - START
          * C            RS       = STAR MEASUREMENT NOISE COVARIANCE
          * C                           (KNOWLEDGE) - START (RAD**2)
          * C            DHCL     = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
          * C                           FROM BORESIGHT - LAMKT (RAD)
          * C            DVCL     = LANDMARK MEASUREMENT VERTICAL DEVIATION
          * C                           FROM BORESIGHT - LAMKT (RAD)
          * C            EDHL     = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
          * C                           DEVIATION FROM BORESIGHT (RAD)
          * C            EDVL     = ESTIMATED LANDMARK MEASUREMENT VERTICAL
          * C                           DEVIATION FROM BORESIGHT (RAD)
          * C            LMU      = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
          * C            RL       = LANDMARK MEASUREMENT NOISE COVARIANCE
          * C                           (KNOWLEDGE) - LAMKT (RAD**2)
          * C
    0050               INCLUDE 'LMTPAR.COM'
    0051  *            COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
          *       .          BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
    0052  *            REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
          *       .          THET
          * C
          * C
          * C      LANDMARK TRACKER PARAMETERS
          * C            AL       = ALTITUDE OF LANDMARK (KM)
          * C            LON      = LONGITUDE OF LANDMARK (DEG)
```

```
        * C              LAT     = LATITUDE OF LANDMARK (DEG)
        * C              TBNL    = ORIENTATION ARRAY FOR LANDMARK TRACKER
        * C                                NOMINAL TO BODY
        * C              TNL     = MISALIGNMENT ARRAY - ACTUAL
        * C                                TRACKER TO NOMINAL
        * C              BL      = BIAS - ACTUAL (RAD)
        * C              SL      = NOISE STANDARD DEVIATION - ACTUAL (RAD)
        * C              BKL     = BIAS - KNOWLEDGE (RAD)
        * C              THET    = LOOK ANGLE (RAD)
        * C              SKL     = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
        * C              TIEO    = INITIAL EARTH FIXED TO INERTIAL
        * C                                TRANSFORMATION
        * C              TNLK    = MISALIGNMENT ARRAY KNOWLEDGE
        * C                                TRACKER TO NOMINAL
        * C              SIGGCP  = POSITION UNCERTAINTY DUE TO CLOUDS
        * C
          C
0053               IF(IENTER .EQ. 1) WRITE(6,1) MCODE
0054      1        FORMAT('     ENTERING EST WITH MCODE = ',I2)
0055               DO 3 I=1,6
0056               XT(I)=X(I)
0057      3        IP(I)=IPN(I)
0058               DO 4 I=7,10
0059               XT(I)=E(I-6)
0060      4        IP(I)=IPA(I-6)
0061               DO 5 I=11,22
0062               IP(I)=IPA(I-6)
0063      5        XT(I)=0.
          C
          C**************************************************************************
          C        CONSTRUCT COMPOSIT STATE TRANSITION MATRIX AND OLD COVARIANCE
          C**************************************************************************
          C
0064               DO 20 I=1,22
0065               DO 20 J=1,22
0066      20       PHI(I,J)=0.
0067               DO 30 I=1,6
0068               DO 30 J=1,6
0069               PO(I,J)=PON(I,J)
0070      30       PHI(I,J)=PN(I,J)
0071               DO 40 I=7,22
0072               DO 40 J=7,22
0073               PO(I,J)=POA(I-6,J-6)
0074      40       PHI(I,J)=PHIA(I-6,J-6)
          C
          C**************************************************************************
          C        ZERO ATTITUDE PORTION OF PROCESS NOISE SINCE IT WAS CONSIDERED
          C        IN THE GYRO MODEL
          C        FORM TOTAL PROCESS NOISE ARRAY
          C**************************************************************************
          C
0075               DO 70 I=1,6
0076               DO 70 J=1,6
0077      70       QT(I,J)=Q(I,J)
          C
          C****************************************************************
          C        ZERO NAV PROCESS NOISE
```

```
          C*************************************************************
          C
0078             DO 75 I=1,6
0079             DO 75 J=1,6
0080      75     Q(I,J)=0.
          C
          C*********************************************************************
          C      PROPAGATE COVARIANCE ARRAY
          C*********************************************************************
          C
0081             CALL PROP(10,22,PHI,COV,PO,QT,P)
          C
          C*********************************************************************
          C      PARTITION INITIALIZED P INTO PN AND PA
          C*********************************************************************
          C
0082             DO 80 I=1,6
0083             DO 80 J=1,6
0084      80     PN(I,J)=P(I,J)
0085             DO 90 I=7,10
0086             DO 90 J=7,10
0087      90     PA(I-6,J-6)=P(I,J)
          C
          C*********************************************************************
          C      ESTABLISH MEASUREMENT PARTIALS
          C*********************************************************************
          C
0088             CALL MPART
          C
          C*********************************************************************
          C      BRANCH TO THE APPROPRIATE UPDATE MODE
          C*********************************************************************
          C
0089             GO TO (100,200,300,400), MCODE
0090             RETURN
          C
          C*********************************************************************
          C      CALCULATE ACTUAL-PREDICTED MEASUREMENT - LANDMARK TRACKER
          C*********************************************************************
          C
0091      100    DZHLM=DHCL-EDHL
0092             DZVLM=DVCL-EDVL
0093             IF(IDEBUG .LT. 4) GO TO 170
0094             WRITE(6,150) DZHLM,DHCL,EDHL
0095             WRITE(6,151) DZVLM,DVCL,EDVL
0096      150    FORMAT('    DZHLM,DHCL,EDHL  ',3E20.10)
0097      151    FORMAT('    DZVLM,DVCL,EDVL  ',3E20.10)
0098             WRITE(6,155)
0099             WRITE(6,152) (PDHL(I),I=1,10)
0100             WRITE(6,153)
0101             WRITE(6,152) (PDVL(I),I=1,10)
0102             WRITE(6,154)
0103             WRITE(6,152) (XT(I),I=1,10)
0104      152    FORMAT(5X,5E15.7)
0105      155    FORMAT('   PDHL')
0106      153    FORMAT('   PDVL')
0107      154    FORMAT('   XT  ')
```

```
C
C****************************************************************
C        MAKE UPDATE - LANDMARK TRACKER
C****************************************************************
C
170      CALL UPDATE(22,10,XT,COV,PO,PDHL,RL(1,1),DZHLM,IP)
         DO 160 I=1,22
         DO 160 J=1,22
160      PO(I,J)=COV(I,J)
         CALL UPDATE(22,10,XT,COV,PO,PDVL,RL(2,2),DZVLM,IP)
         GO TO 600
C
C****************************************************************
C        CALCULATE ACTUAL-PREDICTED MEASUREMENT - GPS
C****************************************************************
C
200      DO 205 I=1,6
205      DXMGPS(I)=MX(I)-EMXG(I)
C
C****************************************************************
C        MAKE UPDATE - GPS
C****************************************************************
C
         CALL UPDATE(22,10,XT,COV,PO,PX,RGPS(1,1),DXMGPS(1),IP)
         DO 210 I=1,22
         DO 210 J=1,22
210      PO(I,J)=COV(I,J)
         Z1=GAIN(1)
         CALL UPDATE(22,10,XT,COV,PO,PY,RGPS(2,2),DXMGPS(2),IP)
         DO 220 I=1,22
         DO 220 J=1,22
220      PO(I,J)=COV(I,J)
         Z2=GAIN(2)
         CALL UPDATE(22,10,XT,COV,PO,PZ,RGPS(3,3),DXMGPS(3),IP)
         DO 230 I=1,22
         DO 230 J=1,22
230      PO(I,J)=COV(I,J)
         Z3=GAIN(3)
         CALL UPDATE(22,10,XT,COV,PO,PXD,RGPS(4,4),DXMGPS(4),IP)
         DO 240 I=1,22
         DO 240 J=1,22
240      PO(I,J)=COV(I,J)
         Z4=GAIN(4)
         CALL UPDATE(22,10,XT,COV,PO,PYD,RGPS(5,5),DXMGPS(5),IP)
         DO 250 I=1,22
         DO 250 J=1,22
250      PO(I,J)=COV(I,J)
         Z5=GAIN(5)
         CALL UPDATE(22,10,XT,COV,PO,PZD,RGPS(6,6),DXMGPS(6),IP)
         Z6=GAIN(6)
         WRITE(13,599)TIME,Z1,Z2,Z3,Z4,Z5,Z6
599      FORMAT(7F14.7)
         GO TO 600
C
C****************************************************************
C        CALCULATE ACTUAL-PREDICTED MEASUREMENT - STAR TRACKER 1
C****************************************************************
```

```
             C
0146   300          DZHST1=DHCS(1)-EDHS(1)
0147                DZVST1=DVCS(1)-EDVS(1)
             C
             C*************************************************************
             C      MAKE UPDATE - STAR TRACKER 1
             C*************************************************************
             C
0148                CALL UPDATE(22,10,XT,COV,PO,PDHS(1,1),RS(1,1,1),DZHST1,IP)
0149                DO 310 I=1,22
0150                DO 310 J=1,22
0151   310          PO(I,J)=COV(I,J)
0152                CALL UPDATE(22,10,XT,COV,PO,PDVS(1,1),RS(2,2,1),DZVST1,IP)
0153                GO TO 600
             C
             C*************************************************************
             C      CALCULATE ACTUAL-PREDICTED MEASUREMENT - STAR TRACKER 2
             C*************************************************************
             C
0154   400          DZHST2=DHCS(2)-EDHS(2)
0155                DZVST2=DVCS(2)-EDVS(2)
             C
             C*************************************************************
             C      MAKE UPDATE - STAR TRACKER 2
             C*************************************************************
             C
0156                CALL UPDATE(22,10,XT,COV,PO,PDHS(1,2),RS(1,1,2),DZHST2,IP)
0157                DO 410 I=1,22
0158                DO 410 J=1,22
0159   410          PO(I,J)=COV(I,J)
0160                CALL UPDATE(22,10,XT,COV,PO,PDVS(1,2),RS(2,2,2),DZVST2,IP)
             C
             C*************************************************************
             C      FINISH
             C*************************************************************
             C
0161   600          DO 650 I=1,6
0162   650          X(I)=XT(I)
0163                DO 700 I=7,10
0164   700          E(I-6)=XT(I)
0165                CALL UNIT(E,E,4)
0166                CALL AMAT(E,A)
0167                CALL CMAT(SF,D,C)
             C
             C*************************************************************
             C      KEEP ATTITUDE COVARIANCE FROM DROPPING TOO LOW
             C*************************************************************
             C
0168                CALL CRAISE(16,COVA,QA,IPA)
0169                IF(IDEBUG .GT. 1) WRITE(6,800) X,E
0170   800          FORMAT('    EXITING FROM EST    ','/,'  X= ',6E20.10,/,
                   .      '    E= ',4E20.10)
0171                IF(IDEBUG .GT. 1) WRITE(6,900) (GAIN(I),I=1,10)
0172   900          FORMAT('  GAIN  ',5E15.7,/,15X,5E15.7)
0173                RETURN
0174                END
```

```
PROGRAM SECTIONS

    Name                              Bytes   Attributes

  0 $CODE                              1631   PIC CON REL LCL   SHR   EXE   RD NOWRT LONG
  1 $PDATA                              232   PIC CON REL LCL   SHR NOEXE   RD NOWRT LONG
  2 $LOCAL                              664   PIC CON REL LCL NOSHR NOEXE   RD   WRT QUAD
  3 DEBUG                                 8   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  4 TARGETS                              36   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  5 PHIA                               7176   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  6 GAIN                                176   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  7 RESIDUALS                            96   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  8 PHIN                               1440   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
  9 UPDT                                752   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 10 ASTATE                              160   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 11 NSTATE                              112   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 12 TIME                                136   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 13 TMAT                                312   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 14 FILTER                               88   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 15 COMPOSIT                          19624   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 16 ARRAYS                             1096   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 17 PART                               2112   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 18 MEASOUT                             648   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG
 19 LMTPAR                              392   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG


ENTRY POINTS

    Address   Type  Name

  0-00000000        EST


VARIABLES

    Address   Type  Name          Address   Type  Name          Address   Type  Name          Address   Type  Name

 19-00000000  R*8   AL         12-00000038  R*8   DATE0      12-00000070  R*8   DATER      12-00000020  R*8   DEL
 18-000001E0  R*8   DHCL       12-00000068  R*8   DTA        12-00000030  R*8   DTN        12-00000080  R*8   DTPRINT
 18-000001E8  R*8   DVCL        7-00000000  R*8   DZHLM       7-00000010  R*8   DZHST1      7-00000020  R*8   DZHST2
  7-00000008  R*8   DZVLM       7-00000018  R*8   DZVST1      7-00000028  R*8   DZVST2     18-00000278  R*8   EDHL
 18-00000280  R*8   EDVL        2-00000018  I*4   I           3-00000004  I*4   IDEBUG      3-00000000  I*4   IENTER
  4-00000004  I*4   IS          2-0000001C  I*4   J           4-0000000C  L*4   JFLAG      19-00000010  R*8   LAT
 19-00000008  R*8   LON         4-00000010  I*4   MCODE       4-00000000  I*4   MTYPE       4-00000008  I*4   NS
  2-00000008  R*8   PDHDX       2-00000010  R*8   PDVDX       4-00000014  R*8   PI          5-00001C00  R*8   QMAX
 11-00000068  R*8   RADE       11-00000060  R*8   RADM       19-00000178  R*8   SIGGCP     19-00000180  R*8   THET
 12-00000018  R*8   TIA        12-00000000  R*8   TIME       12-00000028  R*8   TIN        12-00000058  R*8   TIS
 12-00000060  R*8   TISN       12-00000048  R*8   TMEAS      12-00000008  R*8   TNEXT       4-0000001C  R*8   TPI
 12-00000078  R*8   TPRINT     12-00000050  R*8   TRACK      12-00000010  R*8   TSTOP      12-00000040  R*8   TZERO
  2-00000000  R*8   VDOT        2-00000020  R*4   Z1          2-00000024  R*4   Z2          2-00000028  R*4   Z3
  2-0000002C  R*4   Z4          2-00000030  R*4   Z5          2-00000034  R*4   Z6
```

ARRAYS

Address	Type	Name	Bytes	Dimensions
13-00000000	R*8	A	72	(3, 3)
13-00000048	R*8	B	72	(3, 3)
19-000000C8	R*8	BKL	16	(2)
19-00000048	R*8	BL	16	(2)
13-00000090	R*8	C	72	(3, 3)
15-00001E40	R*8	COV	3872	(22, 22)
5-00000C00	R*8	COVA	2048	(16, 16)
8-00000360	R*8	COVN	288	(6, 6)
10-00000070	R*8	D	24	(3)
10-00000088	R*8	DD	24	(3)
10-00000000	R*8	DE	32	(4)
18-00000150	R*8	DHCS	16	(2)
18-00000160	R*8	DVCS	16	(2)
7-00000030	R*8	DXMGPS	48	(6)
10-00000020	R*8	E	32	(4)
18-00000258	R*8	EDHS	16	(2)
18-00000268	R*8	EDVS	16	(2)
13-000000D8	R*8	EM	96	(4, 3)
18-00000228	R*8	EMXG	48	(6)
6-00000000	R*8	GAIN	176	(22)
15-00003C80	I*4	IP	88	(22)
14-00000018	I*4	IPA	64	(16)
14-00000000	I*4	IPN	24	(6)
18-000001F0	R*8	LMU	24	(3)
18-00000170	R*8	MS	48	(3, 2)
18-00000000	R*8	MX	48	(6)
15-00003D88	R*8	P	3872	(22, 22)
5-00000000	R*8	PA	128	(4, 4)
5-00000200	R*8	PDA	512	(4, 16)
17-000006E0	R*8	PDHL	176	(22)
17-00000420	R*8	PDHS	352	(22, 2)
8-00000120	R*8	PDN	288	(6, 6)
17-00000790	R*8	PDVL	176	(22)
17-00000580	R*8	PDVS	352	(22, 2)
15-00000000	R*8	PHI	3872	(22, 22)
5-00000400	R*8	PHIA	2048	(16, 16)
8-00000240	R*8	PHIN	288	(6, 6)
8-00000000	R*8	PN	288	(6, 6)
15-00002D60	R*8	PO	3872	(22, 22)
5-00001400	R*8	POA	2048	(16, 16)
8-00000480	R*8	PON	288	(6, 6)
17-00000000	R*8	PX	176	(22)
17-00000210	R*8	PXD	176	(22)
17-000000B0	R*8	PY	176	(22)
17-000002C0	R*8	PYD	176	(22)
17-00000160	R*8	PZ	176	(22)
17-00000370	R*8	PZD	176	(22)
9-00000080	R*8	Q	288	(6, 6)
9-00000030	R*8	QA	128	(16)
9-000001D0	R*8	QDOT	288	(6, 6)
9-00000000	R*8	QN	48	(6)
15-00000F20	R*8	QT	3872	(22, 22)
18-00000030	R*8	RGPS	288	(6, 6)

```
18-00000208   R*8   RL          32  (2, 2)
18-000001A0   R*8   RS          64  (2, 2, 2)
10-00000058   R*8   SF          24  (3)
19-000000D8   R*8   SKL         16  (2)
19-000000B8   R*8   SL          16  (2)
16-00000000   R*8   T1          24  (3)
16-00000098   R*8   T11         72  (3, 3)
16-00000018   R*8   T2          24  (3)
16-00000030   R*8   T3          24  (3)
16-000000E0   R*8   T33         72  (3, 3)
16-00000048   R*8   T4          80  (10)
16-00000128   R*8   T44        128  (4, 4)
16-000003E8   R*8   T5          32  (4)
16-00000408   R*8   T6          32  (4)
16-000001A8   R*8   T66        288  (6, 6)
16-00000428   R*8   T7          32  (4)
16-000002C8   R*8   T77        288  (6, 6)
 5-00000080   R*8   TA         384  (4, 12)
19-00000018   R*8   TBNL        72  (3, 3)
19-00000130   R*8   TIED        72  (3, 3)
19-00000060   R*8   TNL         72  (3, 3)
19-000000E8   R*8   TNLK        72  (3, 3)
10-00000040   R*8   WD          24  (3)
11-00000030   R*8   X           48  (6)
11-00000000   R*8   XD          48  (6)
15-00003CD8   R*8   XT         176  (22)
```

A-224

LABELS

```
Address      Label   Address      Label   Address      Label   Address      Label   Address      Label   Address      Label

1-00000000   1'      **           3       **           4       **           5       **           20      **           30
**           40      **           70      **           75      **           80      **           90      0-000001C8   100
1-00000023   150'    1-00000040   151'    1-0000005D   152'    1-0000006F   153'    1-00000079   154'    1-00000065   155'
**           160     0-0000032C   170     0-00000368   200     **           205     **           210     **           220
**           230     **           240     **           250     0-000004EB   300     **           310     0-0000054A   400
**           410     1-00000083   599'    0-000005A7   600     **           650     **           700     1-00000089   800'
1-000000C0   900'
```

FUNCTIONS AND SUBROUTINES REFERENCED

```
AMAT     CMAT     CRAISE     MPART     PROP     UNIT     UPDATE
```

Total Space Allocated = 36891 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

```
COMPILATION STATISTICS

  Run Time:             6.96 seconds
  Elapsed Time:         121.03 seconds
  Page Faults:          438
  Dynamic Memory:       160 pages
```

2.7.1 Propagate Covariances Module (PROP)

The interdependence of the state parameters cause their uncertainty to grow with time. This growth in uncertainty (covariance matrix) is propagated to the measurement time by using the state transition matrix.

The mathematical equation which is used to update the covariance matrix to the time of the measurement is

$$P(t_m) = \phi(t_m, t_{m-1}) \cdot P(t_{m-1}) \cdot \phi(t_m, t_{m-1})^T + Q$$

where

$P(t_m)$ = covariance matrix at time of measurement.
$P(t_{m-1})$ = covariance matrix after processing previous measurement.
$\phi(t_{m-1}, t_{m-1})$ = state transition matrix from time of previous measurement to time of present measurment.
Q - noise matrix to account for model errors.

Figure A-36 illustrates this process

PROPAGATE COVARIANCE (PROP)

I = 1

DOWHILE I .LT. NUMBER OF ROWS IN P (COVARIANCE MATRIX)

COMPUTE Ith ROW IN $P=PHI*P*PHI^T$

I = I + 1

ADD PROCESS NOISE ONTO DIAGONAL
ELEMENTS OF P MATRIX

STORE P IN NEW COVARIANCE MATRIX

INITIALIZE PHI (STATE TRANSITION MATRIX)
TO ZERO WITH ONE'S ON DIAGONAL

Figure A-36

```
0001          SUBROUTINE PROP(NS,N,PHI,PN,PO,Q,PHIP)
         C     DIMENSION PHI(N,N),PN(N,N),PO(N,N),Q(N,N),PHIP(N,N),B(22)
0002            DIMENSION PHI(22,22),PN(22,22),PO(22,22),Q(22,22),PHIP(22,22),B(22)
0003          REAL*8 B,PHI,PHIP,PN,PO,Q
                                                  ! WAS PHIP(NS,NS)
         C*****************************************************************
         C          N      TOTAL NUMBER OF ALLOWABLE PARAMETERS
         C          NS     NUMB R OF STATE PARAMETERS
         C          PHI    STAT  TRANSITION MATRIX
         C          Q      PROC SS NOISE
         C          PN     NEW COVARIANCE MATRIX
         C          PO     OLD COVARIANCE MATRIX
         C
         C
         C            NOTE: PN AND PO ARE THE SAME AT EXIT AND EQUAL TO
         C                  THE UPDATED COVARIANCE. PO IS DISTROYED
         C                  BY INTERNAL PROCESSING.
         C
         C
         C
         C      COVARIANCE PROPAGATION
         C*****************************************************************
0004          INCLUDE 'DEBUG.COM'
0005     *    COMMON /DEBUG/ IENTER,IDEBUG
         * C
         * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
         * C
         * C                 I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
         * C                 IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
         * C
0006          DO 20 I=1,N
0007          DO 10 J=1,N
0008          B(J) = 0.
0009          DO 10 K=1,N
0010      10  B(J) = B(J)+PHI(I,K)*PN(K,J)
0011          DO 20 L=1,N
0012          PO(I,L) = 0.
0013          DO 20 M=1,N
0014      20  PO(I,L) = PO(I,L)+B(M)*PHI(L,M)
0015          DO 30 I=1,N
0016          DO 30 J=1,N
0017          IF (I.LE.NS.AND.J.LE.NS) PO(I,J) = PO(I,J) + Q(I,J)
0018      30  PN(I,J) = PO(I,J)
0019          IF (IDEBUG.GT.2) WRITE(6,25) ((Q(I,J),J=1,6),I=1,6)
0020      25  FORMAT(5X,'Q MATRIX IN PROP:'/(10X,6E15.8))
         C*****************************************************************
         C                INITIALIZE STATE TRANSITION MATRIX
         C*****************************************************************
0021          DO 40 I=1,N          ! WAS NS
0022          DO 40 J=1,N         ! WAS NS
0023          PHIP(I,J) = 0.
0024      40  PHIP(J,J) = 1.
0025          DO 50 I = 1,N
0026               DO 50 J = 1,N
0027      50            Q(I,J) = 0.
         C
         C
```

```
05000   0028          RETURN
05100   0029          END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	526	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	33	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	308	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		PROP

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000080	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	2-00000084	I*4	J
2-00000088	I*4	K	2-0000008C	I*4	L	2-000000C0	I*4	M	AP-00000008@	I*4	N
AP-00000004@	I*4	NS									

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*8	B	176	(22)
AP-0000000C@	R*8	PHI	3872	(22, 22)
AP-0000001C@	R*8	PHIP	3872	(22, 22)
AP-00000010@	R*8	PN	3872	(22, 22)
AP-00000014@	R*8	PO	3872	(22, 22)
AP-00000018@	R*8	Q	3872	(22, 22)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	**	20	1-00000000	25'	**	30	**	40	**	50

Total Space Allocated = 875 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           1.99 seconds
  Elapsed Time:       29.30 seconds
  Page Faults:        353
  Dynamic Memory:     160 pages
```

2.7.2 Compute Measurement Partials (MPART)

Subroutine MPART, called by EST, acts as an executive for the computation of the partials required by the Carlson Filter routine in UPDATE. This routine calls the routines HLMT, HGPS, or HSTAR depending upon the measurement to be made. This measurement is dictated by MCODE an integer flag passed through common.

MPART VCLR

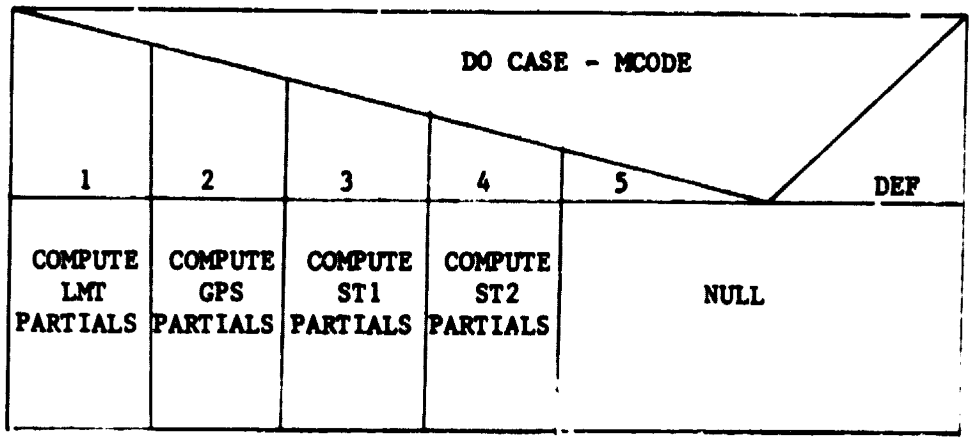


Figure A-37

```
0001                SUBROUTINE MPART
           C**************************************************************
           C
           C        SUBROUTINE MPART GENERATES THE PARTIAL DERIVATIVES OF
           C        THE MEASUREMENT WITH RESPECT TO THE ESTIMATED STATE.
           C        IT ACTS AS AN EXECUTIVE CALLING OTHER SUBROUTINES
           C        TO PERFORM TH  NECESSARY FUNCTIONS.
           C
           C        INPUT PARAMET RS
           C                MCODE   = THE EVENT CODE DETERMINING THE
           C                                MEASUREMENT BEING MADE
           C        OUTPUT PARAMETERS
           C                THE R SPECTIVE PARTIALS FOR THE MEASUREMENT
           C                        BEING MADE THROUGH COMMON
           C        SUBROUTINES CALLED
           C                HLMT    TO COMPUTE THE PARTIALS FOR THE LANDMARK
           C                                TRACKER
           C                HGPS    TO COMPUTE THE PARTIALS FOR THE GPS
           C                HSTAR   TO COMPUTE THE PARTIALS FOR THE KTH
           C                                STAR TRACKER
           C
           C**************************************************************
           C        COMMON BLOCKS
0002                INCLUDE 'TARG TS.COM'
0003     *        COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
0004     *        LOGICAL JFLAG
0005     *        REAL*8 PI,TPI
         * C
         * C              MEASUR MENT SPECIFICATIONS
         * C
         * C                   MTYPE     MEASUREMENT TYPE
         * C                   JFLAG     SET FOR STAR OBSTRUCTION
         * C                   MCODE       "    "  MEASUREMENT PROCESSING
         * C                   PI        PI
         * C                   TPI       2*PI
         * C
           C**************************************************************
           C        BRANCH TO THE APPROPRIATE MEASUREMENT SUBROUTINE
           C**************************************************************
0006                GO TO (10,20,30,40),MCODE
0007                RETURN
           C**************************************************************
           C        COMPUTE THE PARTIALS FOR THE LANDMARK TRACKER
           C**************************************************************
0008       10       CALL HLMT
0009                RETURN
           C**************************************************************
           C        COMPUTE THE PARTIALS FOR THE GPS MEASUREMENTS
           C**************************************************************
0010       20       CALL HGPS
0011                RETURN
           C**************************************************************
           C        COMPUTE THE PARTIALS FOR STAR TRACKER NUMBER ONE
           C**************************************************************
0012       30       CALL HSTAR(1)
0013                RETURN
           C**************************************************************
```

```
             C       COMPUTE THE PARTIALS FOR STAR TRACKER NUMBER TWO
             C*************************************************************
      0014   40      CALL HSTAR(2)
      0015           RETURN
      0016           END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	59	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	8	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	16	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	TARGETS	36	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		MPART

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
3-00000004	I*4	IS	3-0000000C	L*4	JFLAG	3-00000010	I*4	MCODE	3-00000000	I*4	MTYPE
3-00000008	I*4	NS	3-00000014	R*8	PI	3-0000001C	R*8	TPI			

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
0-0000001A	10	0-00000022	20	0-0000002A	30	0-00000032	40

FUNCTIONS AND SUBROUTINES REFERENCED

HGPS HLMT HSTAR

Total Space Allocated = 119 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           0.79 seconds
  Elapsed Time:       15.32 seconds
  Page Faults:        301
  Dynamic Memory:     160 pages
```

2.7.2.1 Compute GPS Measurement Partials (HGPS)

HGPS, Called by MPART, computes the sensitivity of the GPS measurements to variations in the elements of the estimated state. This routine doesn't require the support of any major routine.

Processing Requirements

The GPS measurement vector is given by

$$\underline{\hat{Z}}_G = \left[\hat{x},\ \hat{y},\ \hat{z},\ \hat{\dot{x}},\ \hat{\dot{y}},\ \hat{\dot{z}}\right]^T$$

The sensitivity of this measurement vector to estimated state elements is given by

$$\frac{\partial \hat{Z}_g}{\partial \hat{x}_i} = 1 \text{ for } \hat{x}_i = \hat{x},\ \hat{y},\ \hat{z},\ \hat{\dot{x}},\ \hat{\dot{y}},\ \hat{\dot{z}}$$

$$= 0 \quad \text{otherwise}$$

HGPS VCLR

SET UP ESTIMATED MEASUREMENT VECTOR
ZERO ARRAYS
CONVERT TO RAW-VECTOR FORM

Figure A-38

```
0001            SUBROUTINE HGPS
          C*********************************************************************
          C
          C     SUBROUTINE HGPS GENERATES THE SPECIFIC PARTIALS
          C     RELATING TO THE GLOBAL POSITIONING SYSTEM MEASUREMENTS.
          C
          C     INPUT PARAMET RS
          C             NONE
          C     OUTPUT PARAMETERS
          C             PX      = X POSITION MEASUREMENT PARTIALS
          C             PY      = Y    "         "          "
          C             PZ      = Z    "         "          "
          C             PXD     = X VELOCITY   "          "
          C             PYD     = Y    "         "          "
          C             PZD     = Z    "         "          "
          C
          C*********************************************************************
          C
          C     COMMON BLOCKS
0002            INCLUDE 'PART.COM'
0003      *     COMMON /PART/ PX(22),PY(22),PZ(22),PXD(22),PYD(22),PZD(22),
          *    .        PDHS(22,2),PDVS(22,2),PDHL(22),PDVL(22)
0004      *     REAL*8 PX,PY,PZ,PXD,PYD,PZD,PDHS,PDVS,PDHL,PDVL
          * C
          * C   PARTIALS OF THE RESPECTIVE MEASUREMENTS MADE
          * C
          * C   FOR GPS
          * C           PX      = PARTIALS OF X POSITION MEASUREMENT
          * C           PY      =    "      "  Y    "        "
          * C           PZ      =    "      "  Z    "        "
          * C           PXD     =    "      "  X VELOCITY    "
          * C           PYD     =    "      "  Y    "        "
          * C           PZD     =    "      "  Z    "        "
          * C   FOR STAR TRACKER K (K IS THE SECOND PARAMETER)
          * C           PDHS    = PARTIALS OF HORIZONTAL DIFLECTION
          * C           PDVS    =    "      "  VERTICAL      "
          * C   FOR LANDMARK TRACKER
          * C           PDHL    = PARTIALS OF HORIZONTAL DIFLECTION
          * C           PDVL    =    "      "  VERTICAL      "
          * C
0005            INCLUDE 'NSTATE.COM'
          * C
0006      *   COMMON /NSTATE/ XD(6),X(6),RADM,RADE
0007      *   REAL*8 XD,X,RADM,RADE
          * C
          * C           POSITION STATE AND CONSIDERED PARAMETERS
          * C
          * C                XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
          * C                X         STATE POSITION PARAMETERS (KM AND KM/SEC)
          * C                RADM      RADIUS OF THE MOON (KM)
          * C                RADE      EARTH DETECTABLE RADIUS (KM)
          * C
          * C
0008            INCLUDE 'MEASOUT.COM'
0009      *     COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
          *    .        RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
          *    .        EDHS(2),EDVS(2),EDHL,EDVL
```

```
0010   *          REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
       *                 EDHS,EDVS,EDHL,EDVL
       * C
       * C
       * C        MEASUREMENT OUTPUT PARAMETERS
       * C               MX      = POSITION/VELOCITY STATE MEASUREMENT - GPS
       * C                                  (KM,KM/SEC)
       * C               EMXG    = ESTIMATED POSITION/VELOCITY STATE
       * C                                  MEASUREMENT - HGPS
       * C               RGPS    = STATE MEASUREMENT NOISE COVARIANCE
       * C                                  (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
       * C               DHCS    = STAR MEASUREMENT HORIZONTAL DEVIATION
       * C                                  FROM BORESIGHT - START (RAD)
       * C               DVCS    = STAR MEASUREMENT VERTICAL DEVIATION
       * C                                  FROM BORESIGHT - START (RAD)
       * C               EDHS    = ESTIMATED STAR MEASUREMENT HORIZONTAL
       * C                                  DEVIATION FROM BORESIGHT (RAD)
       * C               EDVS    = ESTIMATED STAR MEASUREMENT VERTICAL
       * C                                  DEVIATION FROM BORESIGHT (RAD)
       * C               MS      = STAR MEASUREMENT UNIT VECTOR (SECOND
       * C                                  SUBSCRIPT REFERS TO TRACKER) - START
       * C               RS      = STAR MEASUREMENT NOISE COVARIANCE
       * C                                  (KNOWLEDGE) - START (RAD**2)
       * C               DHCL    = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
       * C                                  FROM BORESIGHT - LAMKT (RAD)
       * C               DVCL    = LANDMARK MEASUREMENT VERTICAL DEVIATION
       * C                                  FROM BORESIGHT - LAMKT (RAD)
       * C               EDHL    = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
       * C                                  DEVIATION FROM BORESIGHT (RAD)
       * C               EDVL    = ESTIMATED LANDMARK MEASUREMENT VERTICAL
       * C                                  DEVIATION FROM BORESIGHT (RAD)
       * C               LMU     = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
       * C               RL      = LANDMARK MEASUREMENT NOISE COVARIANCE
       * C                                  (KNOWLEDGE) - LAMKT (RAD**2)
       * C
0011              DIMENSION HG(6,10)
0012              REAL*8 HG
         C
         C*************************************************************
         C        SET UP ESTIMATED MEASUREMENT VECTOR
         C*************************************************************
0013              DO 5 I=1,6
0014     5        EMXG(I)=X(I)
         C
         C*************************************************************
         C        ZERO ARRAYS
         C*************************************************************
         C
0015              DO 10 I=1,6
0016              DO 10 J=1,10
0017     10       HG(I,J)=0.
         C
         C*************************************************************
         C        SET UP PARTIALS IN MATRIX FORM
         C*************************************************************
         C
0018              DO 20 I=1,6
```

```
0019    20      HG(I,I)=1.
                C
                C*************************************************************
                C       CONVERT TO ROW VECTOR FORM
                C*************************************************************
                C
0020            DO 30 I=1,10
0021            PX(I)=HG(1,I)
0022            PY(I)=HG(2,I)
0023            PZ(I)=HG(3,I)
0024            PXD(I)=HG(4,I)
0025            PYD(I)=HG(5,I)
0026    30      PZD(I)=HG(6,I)
0027            DO 40 I=11,22
0028            PX(I)=0.
0029            PY(I)=0.
0030            PZ(I)=0.
0031            PXD(I)=0.
0032            PYD(I)=0.
0033    40      PZD(I)=0.
0034            RETURN
0035            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	190	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	488	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 PART	2112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 MEASOUT	648	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		HGPS

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-000001E0	R*8	DHCL	5-000001E8	R*8	DVCL	5-00000278	R*8	EDHL	5-00000280	R*8	EDVL
2-000001E0	I*4	I	2-000001E4	I*4	J	4-00000068	R*8	RADE	4-00000060	R*8	RADM

ARRAYS

Address	Type	Name	Bytes	Dimensions
5-00000150	R*8	DHCS	16	(2)
5-00000160	R*8	DVCS	16	(2)
5-00000258	R*8	EDHS	16	(2)
5-00000268	R*8	EDVS	16	(2)
5-00000228	R*8	EMXG	48	(6)
2-00000000	R*8	HG	480	(6, 10)
5-000001F0	R*8	LMU	24	(3)
5-00000170	R*8	MS	48	(3, 2)
5-00000000	R*8	MX	48	(6)
3-000006E0	R*8	PDHL	176	(22)
3-00000420	R*8	PDHS	352	(22, 2)
3-00000790	R*8	PDVL	176	(22)
3-00000580	R*8	PDVS	352	(22, 2)
3-00000000	R*8	PX	176	(22)
3-00000210	R*8	PXD	176	(22)
3-00000080	R*8	PY	176	(22)
3-000002C0	R*8	PYD	176	(22)
3-00000160	R*8	PZ	176	(22)
3-00000370	R*8	PZD	176	(22)
5-00000030	R*8	RGPS	288	(6, 6)
5-00000208	R*8	RL	32	(2, 2)
5-000001A0	R*8	RS	64	(2, 2, 2)
4-00000030	R*8	X	48	(6)
4-00000000	R*8	XD	48	(6)

A-240

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	5	**	10	**	20	**	30	**	40

Total Space Allocated = 3550 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 1.42 seconds
Elapsed Time: 16.58 seconds
Page Faults: 321
Dynamic Memory: 160 pages

2.7.2.2 COMPUTE LANDMARK TRACKER MEASUREMENT PARTIALS (HLMT)

HLMT, called by MPART, computes the sensitivity of the landmark tracker measurement vector sensitivity to the individual elements of the estimated state vector. This is done with the support of subroutine SPETBI, SPEMI, and PMEAS.

Processing Requirements

A measurement vector from the spacecraft to the landmark is developed by subtracting the estimated spacecraft position ($\hat{\underline{P}}$) vector from the lankmark position vector($\underline{L}$). The resulting measurement vector is transformed to landmark tracker coordinates ($\hat{\underline{M}}_\ell$) and converted to a unit vector ($\hat{\underline{U}}_\ell$). The partials of this unit vector with respect to the estimated state vector elements are given by

$$\frac{\partial \hat{\underline{U}}_\ell}{\partial \hat{x}_i} = \frac{|\hat{\underline{M}}_\ell| \frac{\partial \hat{\underline{M}}_\ell}{\partial \hat{x}_i} - \hat{\underline{M}}_\ell \frac{\partial |\hat{\underline{M}}_\ell|}{\partial \hat{x}_i}}{|\hat{\underline{M}}_\ell|^2}$$

where the terms of the equation are computed by

$$|\hat{\underline{M}}_\ell| = (\hat{\underline{M}}_\ell^T \hat{\underline{M}}_\ell)^{\frac{1}{2}}$$

$$\frac{\partial \hat{\underline{M}}_\ell}{\partial \hat{x}_i} = \{a_{ij}\}\left[\frac{\partial}{\partial \hat{x}_i}\{q_{ij}\}(\underline{L} - \hat{\underline{P}}) + \{q_{ij}\}\frac{\partial}{\partial \hat{x}_i}(\underline{L} - \hat{\underline{P}})\right] \quad , i = 1,2, \ldots, 10$$

$$= 0 \text{ otherwise}$$

$$\frac{\partial |\hat{\underline{M}}_\ell|}{\partial \hat{x}_i} = (\hat{\underline{M}}_\ell^T \frac{\partial}{\partial \hat{x}_i} \hat{\underline{M}}_\ell) |\hat{\underline{M}}_\ell|^{-1} \quad , i = 1, 2, \ldots, 10$$

$$= 0 \text{ otherwise}$$

where

$\hat{\underline{M}}_\ell$	=	the estimated measurement vector in landmark tracker coordinates
$\{a_{ij}\}$	=	transformation from body to landmark tracker coordinates
$\{q_{ij}\}$	=	transformation from inertial body coordinates.
$\hat{\underline{P}}$	=	Spacecraft estimated position vector in inertial coordinates
$\underline{L}$	=	landmark position vector in inertial coordinates.
$\hat{x}_i$	=	estimated ith estimation state vector element.

HLMT VCLR

- Compute LM position vector-inertial space
- Compute estimated measurement-inertial space
- Transform estimated measurement to LMT coordinates
- Convert estimated measurement to unit vector
    - Compute partial of estimated inertial to body transformation WRT estimated state (SPETBI)
    - Compute partial of estimated measurement vector in inertial space WRT estimated state (SPEMI)
    - Compute partial of estimated measurement vector in LMT coordinates WRD estimated state
    - Compute partial of magnitude estimated measurement in LMT coordinates WRD estimated state
    - Compute partial of estimated unit vector in LMT coordinates WRT estimated state
    - Convert unit vector partials to measurement partials (PMEAS)
- Compute measurement partials for states 1 through 10
- Fill all other measurement partial elements (11 through 22) with 0

Figure A-39

```
0001            SUBROUTINE HLMT
          C**********************************************************
          C
          C     SUBROUTINE HLMT COMPUTES THE PARTIALS OF THE LANDMARK
          C     TRACKER MEASUREMENT VECTOR WITH RESPECT TO THE ESTIMATED
          C     STATE VECTOR.
          C
          C     INPUT PARAMETERS
          C             RE      = AVERAGE RADIUS OF THE EARTH
          C             AL      = ALTITUDE OF THE LANDMARK ABOVE RE
          C             LON     = LONGITUDE OF THE LANDMARK
          C             LAT     = LATITUDE  "    "    "
          C             T       = PRESENT TIME
          C             TO      = INITIAL TIME
          C             TIEO    = INITIAL EARTH FIXED TO INERTIAL
          C                           TRANSFORMATION
          C             TBNL    = LANDMARK TRACKER TRANSFORMATION FROM
          C                           NOMINAL TO BOBY
          C             TNLK    = KNOWLEDGE TRANSFORMATION FROM LANDMARK
          C                           TRACKER TO NOMINAL (MISALIGNMENT
          C                           ARRAY)
          C             E       = ESTIMATED QUATERNIAN ARRAY
          C             P       = ESTIMATED POSITION VECTOR IN INERTIAL
          C                           SPACE
          C     OUTPUT PARAMETERS
          C             PDHL    = PARTIALS OF HORIZONTAL DIFLECTION
          C             PDVL    =    "      "  VERTICAL       "
          C
          C**********************************************************
          C
          C     COMMON BLOCKS
0002            INCLUDE 'TIME.COM'
        * C
0003    *       COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
        *      .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0004    *       REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
        *      .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
        * C
        * C             THESE ARE THE TIME REFERENCE FRAMES
        * C
        * C                     TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
        * C                     TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
        * C                     TSTOP     RUN TERMINATION TIME (SEC)
        * C                     TIA       ATTITUDE INTEGRATION TIME (SEC)
        * C                     DEL         "         "        STEP SIZE (SEC)
        * C                     TIN       POSITION INTEGRATION TIME (SEC)
        * C                     DTN         "         "        STEP SIZE (SEC)
        * C                     DATEO     DATE OF FLIGHT EPOCH (JD)
        * C                     DATER     DATE OF 1950 EPOCH (JD)
        * C                     TZERO     START TIME IN SECS. SINCE DATEO
        * C                     TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
        * C                     TIS       REAL WORLD REFERENCE TIME (SEC)
        * C                     TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
        * C                     DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
        * C                               TOO LARGE AT MEASUREMENT TIME
        * C                     TPRINT    TIME FOR PRINT (SEC)
        * C                     DTPRINT   INCREMENT ON TPRINT (SEC)
```

```
              * C
        0005          INCLUDE 'ASTATE.COM'
              * C
        0006  *     COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
        0007  *     REAL*8 DE,E,WD,SF,D,DD
              * C
              * C            ATTITUDE STATE AND CONSIDERED PARAMETERS
              * C
              * C                 D          DIFERENTIAL OF QUATERNIONS
              * C                 E          QUATERNIONS
              * C                 WD         GYRO DRIFT RATE (RAD/SEC)
              * C                 SF         GYRO SCALE FACTOR
              * C                 D          GYRO NON-ORTHOGANALITY (RAD)
              * C                 DD         GYRO RELATIVE ORIENTATION (RAD)
              * C
        0008          INCLUDE 'NSTATE.COM'
              * C
        0009  *     COMMON /NSTATE/ XD(6),X(6),RADM,RADE
        0010  *     REAL*8 XD,X,RADM,RADE
              * C
              * C            POSITION STATE AND CONSIDERED PARAMETERS
              * C
              * C                 XD         STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
              * C                 X          STATE POSITION PARAMETERS (KM AND KM/SEC)
              * C                 RADM       RADIUS OF THE MOON (KM)
              * C                 RADE       EARTH DETECTABLE RADIUS (KM)
              * C
              * C
        0011          INCLUDE 'LMTPAR.COM'
        0012  *       COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
              *     .          BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
        0013  *       REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
              *     .          THET
              * C
              * C
              * C     LANDMARK TRACKER PARAMETERS
              * C              AL        = ALTITUDE OF LANDMARK (KM)
              * C              LON       = LONGITUDE OF LANDMARK (DEG)
              * C              LAT       = LATITUDE OF LANDMARK (DEG)
              * C              TBNL      = ORIENTATION ARRAY FOR LANDMARK TRACKER
              * C                                NOMINAL TO BODY
              * C              TNL       = MISALIGNMENT ARRAY - ACTUAL
              * C                                TRACKER TO NOMINAL
              * C              BL        = BIAS - ACTUAL (RAD)
              * C              SL        = NOISE STANDARD DEVIATION - ACTUAL (RAD)
              * C              BKL       = BIAS - KNOWLEDGE (RAD)
              * C              THET      = LOOK ANGLE (RAD)
              * C              SKL       = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
              * C              TIEO      = INITIAL EARTH FIXED TO INERTIAL
              * C                                TRANSFORMATION
              * C              TNLK      = MISALIGNMENT ARRAY KNOWLEDGE
              * C                                TRACKER TO NOMINAL
              * C              SIGGCP    = POSITION UNCERTAINTY DUE TO CLOUDS
              * C
        0014          INCLUDE 'PART.COM'
        0015  *       COMMON /PART/ PX(22),PY(22),PZ(22),PXD(22),PYD(22),PZD(22),
              *     .          PDHS(22,2),PDVS(22,2),PDHL(22),PDVL(22)
```

```
        REAL*8 PX,PY,PZ,PXD,PYD,PZD,PDHS,PDVS,PDHL,PDVL
C
C       PARTIALS OF THE RESPECTIVE MEASUREMENTS MADE
C
C       FOR GPS
C               PX      = PARTIALS OF X POSITION MEASUREMENT
C               PY      =    "     "  Y    "          "
C               PZ      =    "     "  Z    "          "
C               PXD     =    "     "  X VELOCITY      "
C               PYD     =    "     "  Y    "          "
C               PZD     =    "     "  Z    "
C        FOR STAR TRACKER K (K IS THE SECOND PARAMETER)
C               PDHS    = PARTIALS OF HORIZONTAL DIFLECTION
C               PDVS    =    "     "  VERTICAL        "
C        FOR LANDMARK TRACKER
C               PDHL    = PARTIALS OF HORIZONTAL DIFLECTION
C               PDVL    =    "     "  VERTICAL        "
C
         INCLUDE 'TARG TS.COM'
        COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
        LOGICAL JFLAG
        REAL*8 PI,TPI
C
C               MEASUR.MENT SPECIFICATIONS
C
C                       MTYPE       MEASUREMENT TYPE
C                       JFLAG       SET FOR STAR OBSTRUCTION
C                       MCODE        "   "  MEASUREMENT PROCESSING
C                       PI          PI
C                       TPI         2*PI
C
         INCLUDE 'DEBUG.COM'
        COMMON /DEBUG/ IENTER,IDEBUG
C
C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
C
C                       I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
C                       IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
C
         INCLUDE 'MEASOUT.COM'
         COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
     *          RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
     *          EDHS(2),EDVS(2),EDHL,EDVL
         REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
     *          EDHS,.DVS,EDHL,EDVL
C
C
C        MEASUREMENT OUTPUT PARAMETERS
C                  MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
C                                  (KM,KM/SEC)
C                  EMXG     = ESTIMATED POSITION/VELOCITY STATE
C                                  MEASUREMENT - HGPS
C                  RGPS     = STATE MEASUREMENT NOISE COVARIANCE
C                                  (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
C                  DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
C                                  FROM BORESIGHT - START (RAD)
C                  DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
```

```
        * C                              FROM BORESIGHT - START (RAD)
        * C                 EDHS    = ESTIMATED STAR MEASUREMENT HORIZONTAL
        * C                              DEVIATION FROM BORESIGHT (RAD)
        * C                 EDVS    = ESTIMATED STAR MEASUREMENT VERTICAL
        * C                              DEVIATION FROM BORESIGHT (RAD)
        * C                 MS      = STAR MEASUREMENT UNIT VECTOR (SECOND
        * C                              SUBSCRIPT REFERS TO TRACKER) - START
        * C                 RS      = STAR MEASUREMENT NOISE COVARIANCE
        * C                              (KNOWLEDGE) - START (RAD**2)
        * C                 DHCL    = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
        * C                              FROM BORESIGHT - LAMKT (RAD)
        * C                 DVCL    = LANDMARK MEASUREMENT VERTICAL DEVIATION
        * C                              FROM BORESIGHT - LAMKT (RAD)
        * C                 EDHL    = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
        * C                              DEVIATION FROM BORESIGHT (RAD)
        * C                 EDVL    = ESTIMATED LANDMARK MEASUREMENT VERTICAL
        * C                              DEVIATION FROM BORESIGHT (RAD)
        * C                 LMU     = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
        * C                 RL      = LANDMARK MEASUREMENT NOISE COVARIANCE
        * C                              (KNOWLEDGE) - LAMKT (RAD**2)
        * C
0026              EQUIVALENCE (RE,RADE),(T,TIME),(EP,X)
0027              DIMENSION LE(3),TIE(3,3),L(3),ETBL(3,3),ETLB(3,3),ETBI(3,3),
           .           EMI(3),ETLI(3,3),EML(3),PETBI(3,3),PEMI(3),EP(3),
           .           DM2(3),DM1(3),PEMB(3),PEML(3),PEUL(3),EUL(3)
0028              REAL*8 MEML,PMEML,TO,DM1,DM2,EMI,EML,ETBI,ETBL,ETLB,ETLI,
           .           EUL,L,LE,PEMB,PEML,PETBI,PEUL,TIE,T,EP,PEMI,VMAG,
           .           RE
          C************************************************************
          C       COMPUTE LE - LANDMARK POSITION VECTOR IN EARTH FIXED
          C                    COORDINATES
          C************************************************************

0029              LE(1)=(RE+AL)*COS(LON*PI/180.)*COS(LAT*PI/180.)
0030              LE(2)=(RE+AL)*SIN(LON*PI/180.)*COS(LAT*PI/180.)
0031              LE(3)=(RE+AL)*SIN(LAT*PI/180.)
          C************************************************************
          C       COMPUTE TIE - EXACT TRANSFORMATION FROM EARTH FIXED TO
          C                    INERTIAL SPACE
          C************************************************************
0032              CALL WET(TIEO,TIE)
          C************************************************************
          C       COMPUTE L - LANDMARK POSITION IN INERTIAL SPACE
          C************************************************************
0033              CALL MATAB(TIE,LE,L,3,3,1)
          C************************************************************
          C       COMPUTE ETLB - ESTIMATED BODY TO LANDMARK TRACKER
          C                    TRANSFORMATION
          C************************************************************
0034              CALL MATAB(TBNL,TNLK,ETBL,3,3,3)
0035              CALL MINV3(ETBL,ETLB)
          C************************************************************
          C       COMPUTE ETBI - ESTIMATED INERTIAL TO BODY TRANSFORMATION
          C************************************************************
0036              CALL AMAT(E,ETBI)
          C************************************************************
          C       COMPUTE EMI - ESTIMATED MEASUREMENT IN INERTIAL SPACE
```

```
              C*****************************************************************
0037                  DO 10 I=1,3
0038          10      EMI(I)=L(I)-EP(I)
              C*****************************************************************
              C       TRANSFORM TO LANDMARK TRACKER COORDINATES
              C*****************************************************************
0039                  CALL MATAB(ETLB,ETBI,ETLI,3,3,3)
0040                  CALL MATAB(ETLI,EMI,EML,3,3,1)
              C*****************************************************************
              C       COMPUTE MEML - MAGNITUDE OF EML
              C*****************************************************************
0041                  MEML=VMAG(EML,3)
              C*****************************************************************
              C       COMPUTE EUL - ESTIMATED UNIT VECTOR ALONG ESTIMATED
              C                     MEASUREMENT VECTOR
              C*****************************************************************
0042                  CALL UNIT(EML,EUL,3)
0043                  EDVL=ASIN(EUL(1))
0044                  EDHL=ASIN(EUL(2)/COS(EDVL))
0045                  IF(IDEBUG .LT. 4)GO TO 15
0046                  WRITE(6,42)
0047                  WRITE(6,12)
0048          12      FORMAT('      ETBI')
0049                  WRITE(6,13)((ETBI(I,J),J=1,3),I=1,3)
0050          13      FORMAT(10X,3E17.7)
0051          15      CONTINUE
              C*****************************************************************
              C       COMPUTE MEASUREMENT PARTIALS
              C*****************************************************************
0052                  DO 100 J=1,10
              C*****************************************************************
              C       COMPUTE PARTIAL OF ETBI WRT ESTIMATED STATE
              C*****************************************************************
0053                  CALL SPETBI(J,PETBI)
              C*****************************************************************
              C       COMPUTE PEMI - PARTIAL OF EMI
              C*****************************************************************
0054                  CALL SPEMI(J,PEMI)
              C*****************************************************************
              C       COMPUTE PEML - PARTIAL OF EML
              C*****************************************************************
0055                  CALL MATAB(ETBI,PEMI,DM2,3,3,1)
0056                  CALL MATAB(PETBI,EMI,DM1,3,3,1)
0057                  DO 20 I=1,3
0058          20      PEMB(I)=DM1(I)+DM2(I)
0059                  CALL MATAB(ETLB,PEMB,PEML,3,3,1)
              C*****************************************************************
              C       COMPUTE PMEML - PARTIAL OF MAGNITUDE OF EML
              C*****************************************************************
0060                  PMEML=0.
0061                  DO 30 I=1,3
0062          30      PMEML=PMEML+(PEML(I)*EML(I)/MEML)
              C*****************************************************************
              C       COMPUTE PEUL - PARTIAL OF ESTIMATED UNIT VECTOR
              C*****************************************************************
0063                  DO 40 I=1,3
0064          40      PEUL(I)=(MEML*PEML(I)-PMEML*EML(I))/(MEML**2)
```

```
              IF(IDEBUG .LT. 4) GO TO 50
              WRITE(6,42)
     42       FORMAT(' FROM HLMT ')
              WRITE(6,43) J
     43       FORMAT('      J= ',I3)
              WRITE(6,44) M ML,(PEML(I),I=1,3)
     44       FORMAT('            MEML,PEML(I) ',4(5X,E15.7))
              WRITE(6,45) PMEML,(EML(I),I=1,3)
     45       FORMAT('            PMEML,EML(I) ',4(5X,E15.7))
              WRITE(6,46) (PEUL(I),I=1,3)
     46       FORMAT('                   PEUL(I) ',20X,3(5X,E15.7))
     50       CONTINUE
     C***************************************************************
     C        CONVERT UNIT VECTOR PARTIALS TO MEASUREMENT PARTIALS
     C***************************************************************
     100      CALL PMEAS(PEUL,EUL,PDHL(J),PDVL(J),2)
              DO 200 I=11,22
              PDHL(I)=0.
     200      PDVL(I)=0.
              RETURN
              END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	812	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	167	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	1052	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	ASTATE	160	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	LMTPAR	392	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	PART	2112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	TARGETS	36	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
9	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
10	MEASOUT	648	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		HLMT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
6-00000000	R*8	AL	3-00000038	R*8	DATEO	3-00000070	R*8	DATER	3-00000020	R*8	DEL
10-000001E0	R*8	DHCL	3-00000068	R*8	DTA	3-00000030	R*8	DTN	3-00000080	R*8	DTPRINT
10-000001E8	R*8	DVCL	10-00000278	R*8	EDHL	10-00000280	R*8	EDVL	2-00000200	I*4	I
9-00000004	I*4	IDEBUG	9-00000000	I*4	IENTER	8-00000004	I*4	IS	2-00000204	I*4	J

8-0000000C	L*4	JFLAG	6-00000010	R*8	LAT	6-00000008	R*8	LON	8-00000010	I*4	MCODE
2-000002B8	R*8	MEML	8-00000000	I*4	MTYPE	8-00000008	I*4	NS	8-00000014	R*8	PI
2-000002C0	R*8	PMEML	5-00000068	R*8	RADE	5-00000060	R*8	RADM	5-00000068	R*8	RE
6-00000178	R*8	SIGGCP	3-00000000	R*8	T	2-000002C8	R*8	TO	6-00000180	R*8	THET
3-00000018	R*8	TIA	3-00000000	R*8	TIME	3-00000028	R*8	TIN	3-00000058	R*8	TIS
3-00000060	R*8	TISN	3-00000048	R*8	TMEAS	3-00000008	R*8	TNEXT	8-0000001C	R*8	TPI
3-00000078	R*8	TPRINT	3-00000050	R*8	TRACK	3-00000010	R*8	TSTOP	3-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
6-000000C8	R*8	BKL	16	(2)
6-000000A8	R*8	BL	16	(2)
4-00000070	R*8	D	24	(3)
4-00000083	R*8	DD	24	(3)
4-00000000	R*8	DF	32	(4)
10-00000150	R*8	DHCS	16	(2)
2-00000240	R*8	DM1	24	(3)
2-00000228	R*8	DM2	24	(3)
10-00000160	R*8	DVCS	16	(2)
4-00000020	R*8	E	32	(4)
10-00000258	R*8	EDHS	16	(2)
10-00000268	R*8	EDVS	16	(2)
2-00000150	R*8	EMI	24	(3)
2-000001B0	R*8	EML	24	(3)
10-00000228	R*8	EMXG	48	(6)
5-00000030	R*8	EP	24	(3)
2-00000108	R*8	ETBI	72	(3, 3)
2-00000078	R*8	ETBL	72	(3, 3)
2-000000C0	R*8	ETLB	72	(3, 3)
2-00000168	R*8	ETLI	72	(3, 3)
2-000002A0	R*8	EUL	24	(3)
2-00000060	R*8	L	24	(3)
2-00000000	R*8	LE	24	(3)
10-000001F0	R*8	LMU	24	(3)
10-00000170	R*8	MS	48	(3, 2)
10-00000000	R*8	MX	48	(6)
7-000006E0	R*8	PDHL	176	(22)
7-00000420	R*8	PDHS	352	(22, 2)
7-00000790	R*8	PDVL	176	(22)
7-00000580	R*8	PDVS	352	(22, 2)
2-00000258	R*8	PEMB	24	(3)
2-00000210	R*8	PEMI	24	(3)
2-00000270	R*8	PEML	24	(3)
2-000001C8	R*8	PETBI	72	(3, 3)
2-00000288	R*8	PEUL	24	(3)
7-00000000	R*8	PX	176	(22)
7-00000210	R*8	PXD	176	(22)
7-000000B0	R*8	PY	176	(22)
7-000002C0	R*8	PYD	176	(22)
7-00000160	R*8	PZ	176	(22)
7-00000370	R*8	PZD	176	(22)
10-00000030	R*8	RGPS	288	(6, 6)
10-00000208	R*8	RL	32	(2, 2)
10-000001A0	R*8	RS	64	(2, 2, 2)

```
4-00000058  R*8  SF        24  (3)
6-000000D8  R*8  SKL       16  (2)
6-000000B8  R*8  SL        16  (2)
6-00000018  R*8  TBNL      72  (3, 3)
2-00000018  R*8  TIE       72  (3, 3)
6-00000130  R*8  TIEO      72  (3, 3)
6-00000060  R*8  TNL       72  (3, 3)
6-000000E8  R*8  TNLK      72  (3, 3)
4-00000040  R*8  WD        24  (3)
5-00000030  R*8  X         48  (6)
5-00000000  R*8  XD        48  (6)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	1-00000000	12'	1-0000000D	13'	0-0000019D	15	**	20	**	30
**	40	1-00000015	42'	1-00000023	43'	1-00000030	44'	1-00000053	45'	1-00000076	46'
0-000002EA	50	**	100	**	200						

FUNCTIONS AND SUBROUTINES REFERENCED

```
AMAT      MATAB     MINV3     MTH$DASIN   MTH$DCOS   MTH$DSIN   PMEAS   SPEMI
SPETBI    UNIT      VMAG      WET
```

A-250

Total Space Allocated = 5635 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
Run Time:          3.58 seconds
Elapsed Time:      56.86 seconds
Page Faults:       429
Dynamic Memory:    160 pages
```

2.7.2.2.1 COMPUTE PARTIALS WITH RESPECT TO ESTIMATED TRANSFORMATION FROM INERTIAL TO BODY COORDATES (SPETBI)

SPETBI, called by HLMT and HSTAR, computes the sensivity of the estimated inertial to body transformation to the individual elements of the estimated state. This routine is essentially unsupported by any other major routines.

Processor Requirements

Recognizing that

$$\{\hat{q}_{ij}\} = \begin{bmatrix} \hat{q}_0^2 + \hat{q}_1^2 - \hat{q}_2^2 - \hat{q}_3^2 & 2(\hat{q}_1\hat{q}_2 + \hat{q}_3\hat{q}_0) & 2(\hat{q}_1\hat{q}_3 - \hat{q}_2\hat{q}_0) \\ 2(\hat{q}_1\hat{q}_2 - \hat{q}_3\hat{q}_0) & \hat{q}_0^2 - \hat{q}_1^2 + \hat{q}_2^2 - \hat{q}_3^2 & 2(\hat{q}_2\hat{q}_3 + \hat{q}_1\hat{q}_0) \\ 2(\hat{q}_1\hat{q}_3 - \hat{q}_2\hat{q}_0) & 2(\hat{q}_2\hat{q}_3 - \hat{q}_1\hat{q}_0) & \hat{q}_0^2 - \hat{q}_1^2 - \hat{q}_2^2 + \hat{q}_3^2 \end{bmatrix}$$

since

$$x_i = q_i - 7 \qquad \text{for } i = 7, 8, 9, \text{ and } 10$$

$$\frac{\partial}{\partial \hat{x}_i}\{q_{ij}\} = 2\begin{bmatrix} \hat{q}_0 & \hat{q}_3 & -\hat{q}_2 \\ -\hat{q}_3 & +\hat{q}_0 & \hat{q}_1 \\ \hat{q}_2 & -\hat{q}_1 & \hat{q}_0 \end{bmatrix} \qquad \text{for } i = 7$$

$$= 2\begin{bmatrix} \hat{q}_1 & \hat{q}_2 & \hat{q}_3 \\ \hat{q}_2 & -\hat{q}_1 & \hat{q}_0 \\ \hat{q}_3 & -\hat{q}_0 & -\hat{q}_1 \end{bmatrix} \qquad \text{for } i = 8$$

$$= 2\begin{bmatrix} -\hat{q}_2 & \hat{q}_1 & -\hat{q}_0 \\ \hat{q}_1 & \hat{q}_2 & \hat{q}_3 \\ \hat{q}_0 & \hat{q}_3 & -\hat{q}_2 \end{bmatrix} \qquad \text{for } i = 9$$

$$= 2\begin{bmatrix} -\hat{q}_3 & \hat{q}_0 & \hat{q}_1 \\ -\hat{q}_0 & -\hat{q}_3 & \hat{q}_2 \\ \hat{q}_1 & \hat{q}_2 & \hat{q}_3 \end{bmatrix} \qquad \text{for } i = 10$$

and

$$\frac{\partial}{\partial \hat{x}_i}\left\{q_{ij}\right\} = \left[\ 0\ \right] \qquad \text{for } i < 7 \text{ and } i > 10$$

SPETBI (I,Q)

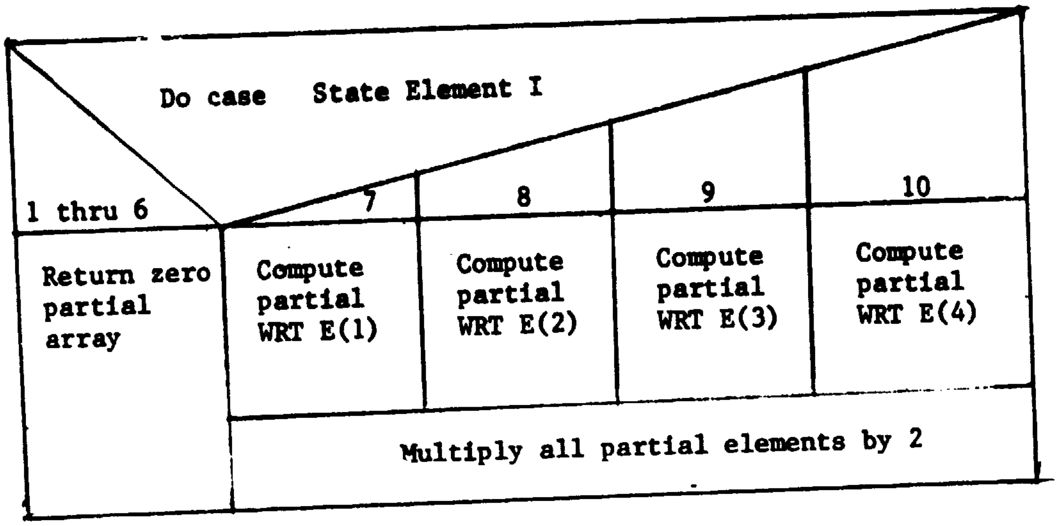


Figure A-40

```
              SUBROUTINE SPETBI(I,Q)
        C*************************************************************
        C
        C     SUBROUTINE SPETBI COMPUTES THE PARTIAL OF THE ESTIMATED
        C     TRANSFORMATION FROM INERTIAL TO BODY COORDINATES.
        C
        C     INPUT PARAMETERS
        C             I        = THE INTEGER CONTROLLING THE SPECIFIC
        C                          PARTIAL BEING COMPUTED
        C                          (RANGE 1-10)
        C             E        = THE ESTIMATED QUATERNIAN ARRAY
        C     OUTPUT PARAMETERS
        C             Q        = THE OUTPUT PARTIAL ARRAY (3,3)
        C     CALLED BY HSTAR AND HLMT
        C
        C*************************************************************
        C
        C     COMMON BLOCKS
              INCLUDE 'ASTATE.COM'
      * C
      *     COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
      *     REAL*8 DE,E,WD,SF,D,DD
      * C
      * C             ATTITUDE STATE AND CONSIDERED PARAMETERS
      * C
      * C                  DE        DIFERENTIAL OF QUATERNIONS
      * C                  E         QUATERNIONS
      * C                  WD        GYRO DRIFT RATE (RAD/SEC)
      * C                  SF        GYRO SCALE FACTOR
      * C                  D         GYRO NON-ORTHOGANALITY (RAD)
      * C                  DD        GYRO RELATIVE ORIENTATION (RAD)
      * C
              DIMENSION Q(3,3)
              REAL*8 Q
        C*************************************************************
        C     BRANCH TO THE APPROPRIATE SECTOR ON I
        C*************************************************************
              GO TO (10,10,10,10,10,10,20,30,40,50),I
        C*************************************************************
        C     ZERO Q ARRAY - POSITION OR VELOCITY ELEMENT PARTIAL
        C*************************************************************
        10    DO 15 J=1,3
              DO 15 K=1,3
        15    Q(J,K)=0.
              RETURN
        C*************************************************************
        C     COMPUTE PARTIALS WRT Q(0)
        C*************************************************************
        20    Q(1,1)=E(1)
              Q(1,2)=E(4)
              Q(1,3)=-E(3)
              Q(2,1)=-E(4)
              Q(2,2)=E(1)
              Q(2,3)=E(2)
              Q(3,1)=E(3)
              Q(3,2)=-E(2)
              Q(3,3)=E(1)
```

```
              GO TO 100
C*****************************************************************
C             COMPUTE PARTIALS WRT Q(1)
C*****************************************************************
      30      Q(1,1)=E(2)
              Q(1,2)=E(3)
              Q(1,3)=E(4)
              Q(2,1)=E(3)
              Q(2,2)=-E(2)
              Q(2,3)=E(1)
              Q(3,1)=E(4)
              Q(3,2)=-E(1)
              Q(3,3)=-E(2)
              GO TO 100
C*****************************************************************
C             COMPUTE PARTIALS WRT Q(2)
C*****************************************************************
      40      Q(1,1)=-E(3)
              Q(1,2)=E(2)
              Q(1,3)=-E(1)
              Q(2,1)=E(2)
              Q(2,2)=E(3)
              Q(2,3)=E(4)
              Q(3,1)=E(1)
              Q(3,2)=E(4)
              Q(3,3)=-E(3)
              GO TO 100
C*****************************************************************
C             COMPUTE PARTIALS WRT Q(3)
C*****************************************************************
      50      Q(1,1)=-E(4)
              Q(1,2)=E(1)
              Q(1,3)=E(2)
              Q(2,1)=-E(1)
              Q(2,2)=-E(4)
              Q(2,3)=E(3)
              Q(3,1)=E(2)
              Q(3,2)=E(3)
              Q(3,3)=E(4)
      100     DO 200 J=1,3
              DO 200 K=1,3
      200     Q(J,K)=2.*Q(J,K)
              RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	349	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	28	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 ASTATE	160	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		SPETBI

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name
AP-00000004@	I*4	I	2-00000000	I*4	J	2-00000004	I*4	K

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-00000070	R*8	D	24	(3)
3-00000088	R*8	DD	24	(3)
3-00000000	R*8	DE	32	(4)
3-00000020	R*8	E	32	(4)
AP-00000008@	R*8	Q	72	(3, 3)
3-00000058	R*8	SF	24	(3)
3-00000040	R*8	WD	24	(3)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-0000002D	10	**	15	0-00000051	20	0-0000008C	30	0-000000C6	40	0-00000100	50
0-00000138	100	**	200								

Total Space Allocated = 537 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

```
  Run Time:             2.07 seconds
  Elapsed Time:         36.65 seconds
  Page Faults:          324
  Dynamic Memory:       160 pages
```

2.7.2.2.2 COMPUTE PARTIALS OF ESTIMATED MEASUREMENT VECTOR IN INERTIAL SPACE (SPEMI)

SPEMI called by HLMT, computes the sensivity of the estimated landmark tracker measurement vector in inertial space with respect to the elements of the estimated state. This routine is unsupported by other major routines.

Processing Requirements

Recognizing that

$$\underline{\hat{M}_I} = \underline{L} - \underline{\hat{P}}$$

$$\frac{\partial \underline{\hat{M}_I}}{\partial \hat{x}_i} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \quad \text{for } \hat{x}_i = \hat{x}$$

$$\frac{\partial \underline{\hat{M}_I}}{\partial \hat{x}_i} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \quad \text{for } \hat{x}_i = \hat{y}$$

$$\frac{\partial \underline{\hat{M}_I}}{\partial \hat{x}_i} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad \text{for } \hat{x}_i = \hat{z}$$

and

$$\frac{\partial \underline{\hat{M}_I}}{\partial \hat{x}_i} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{otherwise}$$

where

$\underline{\hat{M}_I}$ = Estimated measurement vector in inertial space

$\underline{L}$ = Landmark position vector in inertial space

$\underline{\hat{P}} = \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix}$ = Spacecraft estimated position vector in inertial space.

SPEMI (I,X)

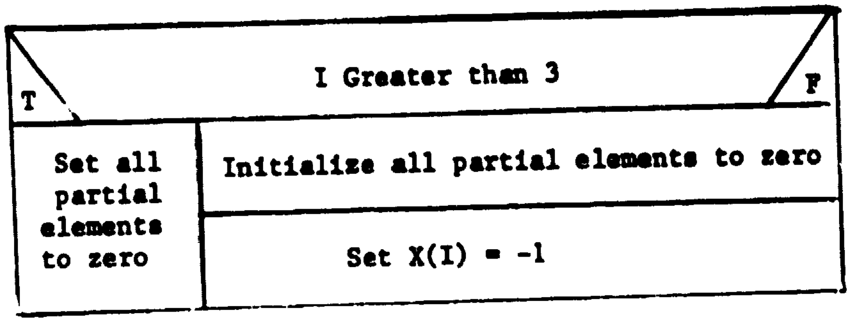


Figure A-41

```
            SUBROUTINE SPEMI(I,X)
      C*************************************************************
      C
      C     SUBROUTINE SPEMI COMPUTES THE PARTIAL OF THE MEASUREMENT
      C     VECTOR IN INERTIAL SPACE WRT THE ESTIMATED STATE
      C
      C     INPUT PARAMETERS
      C           I       = THE INTEGER CONTROLLING THE INDIVIDUAL
      C                         PARTIAL (RANGE 1-10)
      C     OUTPUT PARAMETERS
      C           X       = THE OUTPUT PARTIAL VECTOR (3,1)
      C     CALLED BY HLMT
      C
      C*************************************************************
            DIMENSION X(3)
            REAL*8 X
            IF(I .GT. 3) GO TO 50
            DO 20 J=1,3
      20    X(J)=0.
            X(I)=-1.
            RETURN
      50    DO 60 J=1,3
      60    X(J)=0.
            RETURN
            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	66	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	24	PIC CON REL LCL NOSHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		SPEMI

VARIABLES

Address	Type	Name	Address	Type	Name
AP-00000004@	I*4	I	2-00000000	I*4	J

SPEMI

6-Apr-1981 15:02:29 VAX-11 FORTRAN V2.0-2
11-Sep-1980 11:56:21 _DBA0:[D11R.GCP]SPEMI.FOR;4

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	X	24	(3)

LABELS

Address	Label	Address	Label	Address	Label
**	20	0-00000034	50	**	60

Total Space Allocated = 90 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 0.56 seconds
Elapsed Time: 11.04 seconds
Page Faults: 306
Dynamic Memory: 160 pages

2.7.2.3 MEASUREMENT PARTIALS FOR STAR TRACKER (K) (HSTAR)

HSTAR, called by MPART, computes the sensivity of the star tracker measurement to variations in the individual elements of the estimated state vector. This is done with the support of subroutines SPETBI and PMEAS.

Processing Requirements

A unit vector along the optical axis at each star tracker (K) is established and transformed to inertial space through the estimated transformation from star tracker coordinates to inertial space ($\underline{U}s$). The partials of this unit vector with respect to the element of the estimated state are given by

$$\frac{\partial \hat{U}s}{\partial \hat{x}_i} = \left[\{b_{ij}\} \frac{\partial}{\partial \hat{x}_i} \{\hat{q}_{ij}\}\{a_{ij}\} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right]$$

where

$\{b_{ij}\}$ = Transformation from body to star tracker coordinates.

$\{\hat{q}_{ij}\}$ = Estimated transformation from inertial to body coordinates.

$\{a_{ij}\}$ = Ideal transformation from star tracker to inertial coordinates.

Compute star unit vector in inertial coordinates

Compute estimated transformation from star tracker to inertial space

Computed estimated unit vector in ST coordinates

Compute estimated measurement

- Compute partial of estimated transformation from inertial to body WRT estimated state (SPETBI)
- Compute partials of estimated unit vector
- Compute unit vector partials to measurement partials

Compute partials of unit vector with respect to states 7 through 10

Fill all other measurement partial elements (1 through 6) and (11 through 22) with zero

Figure A-42

```
0001                SUBROUTINE HSTAR(K)
          C*****************************************************************
          C
          C         SUBROUTINE HSTAR COMPUTES THE PARTIALS FOR THE KTH STAR
          C         TRACKER MEASUREMENT VECTOR.
          C
          C         INPUT PARAMET RS
          C                  K        = SELECTOR OF THE KTH STAR TRACKER
          C                  STATE    = ELEMENTS 7-10 ARE USED AS THE IDEAL
          C                                  INERTIAL TO BODY QUATERNIAN ELEMENTS
          C                  TBNS     = STAR TRACKER NOMINAL TO BODY TRANSFORMATION
          C                  TNS      =  "        "      STAR TRACKER TO NOMINAL
          C                                  TRANSFORMATION (MISALIGNMENT ARRAY)
          C                  E        = ESTIMATED QUATERNIAN ARRAY
          C         OUTPUT PARAMETERS
          C                  PDHS     = PARTIAL OF THE STAR TRACKER HORAZONTAL
          C                                  DEFLECTION
          C                  PDVS     = PARTIAL OF THE STAR TRACKER VERTICAL
          C                                  DEFLECTION
          C
          C*****************************************************************
          C
          C         COMMON BLOCKS
                    INCLUDE 'PART.COM'
0002                COMMON /PART/ PX(22),PY(22),PZ(22),PXD(22),PYD(22),PZD(22),
0003      *        .        PDHS(22,2),PDVS(22,2),PDHL(22),PDVL(22)
          *         REAL*8 PX,PY,PZ,PXD,PYD,PZD,PDHS,PDVS,PDHL,PDVL
0004      * C
          * C       PARTIALS OF THE RESPECTIVE MEASUREMENTS MADE
          * C
          * C       FOR GPS
          * C                PX       = PARTIALS OF X POSITION MEASUREMENT
          * C                PY       =    "      "  Y     "          "
          * C                PZ       =    "      "  Z     "          "
          * C                PXD      =    "      "  X VELOCITY       "
          * C                PYD      =    "      "  Y     "          "
          * C                PZD      =    "      "  Z     "          "
          * C       FOR STAR TRACKER K (K IS THE SECOND PARAMETER)
          * C                PDHS     = PARTIALS OF HORIZONTAL DIFLECTION
          * C                PDVS     =    "      "  VERTICAL         "
          * C       FOR LANDMARK TRACKER
          * C                PDHL     = PARTIALS OF HORIZONTAL DIFLECTION
          * C                PDVL     =    "      "  VERTICAL         "
          * C
                    INCLUDE 'STARPAR.COM'
0005                COMMON /STARPAR/ BS(2,2),SS(2,2),TNS(3,3,2),TBNS(3,3,2),
0006      *        .        BSK(2,2),SSK(2,2),TNSK(3,3,2)
          *         REAL*8 BS,SS,TNS,TBNS,BSK,SSK,TNSK
0007      * C
          * C       STAR TRACKER PARAMETERS
          * C                IN EACH CASE THE LAST SUBSCRIPT REFERS TO THE
          * C                         TRACKER USED
          * C                BS       = BIAS - ACTUAL (RAD)
          * C                SS       = NOISE STANDARD DEVIATION - ACTUAL (RAD)
          * C                TNS      = MISALIGNMENT ARRAY - TRANSFORMATION FROM
          * C                                 STAR TRACKER TO NOMINAL
          * C                TBNS     = ORIENTATION ARRAY - TRANSFORMATION FROM
```

```
02600          * C                             NOMINAL TO BODY
02600          * C                BSK      = BIAS - KNOWLEDGE (RAD)
02600          * C                SSK      = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
02600          * C                TNSK     = MISALIGNMENT KNOWLEDGE ARRAY - TRANSFORMATION
02600          * C                             FROM STAR TRACKER TO NOMINAL
02600          * C
02600   0008          INCLUDE 'ENVIR.COM'
02700   0009   *     COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
02700   0010   *     REAL*8 STATE,PROFILE
02700          * C
02700          * C             REAL WORLD STATE PARAMETERS
02700          * C
02700          * C                  STATE     STATE VALUES: X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
02700          * C                  PROFILE   ATTITUDE PROFILE-TIME (SEC) VS
02700          * C                            INERTIAL ANGULAR RATES (RAD/SEC)
02700          * C                  INIT      INTEGRATION INITIALIZATION KEY (-1)
02700          * C
02700   0011          INCLUDE 'ASTATE.COM'
02800          * C
02800   0012   *     COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
02800   0013   *     REAL*8 DE,E,WD,SF,D,DD
02800          * C
02800          * C             ATTITUDE STATE AND CONSIDERED PARAMETERS
02800          * C
02800          * C                  D_        DIFERENTIAL OF QUATERNIONS
02800          * C                  E         QUATERNIONS
02800          * C                  WD        GYRO DRIFT RATE (RAD/SEC)
02800          * C                  SF        GYRO SCALE FACTOR
02800          * C                  D         GYRO NON-ORTHOGANALITY (RAD)
02800          * C                  DD        GYRO RELATIVE ORIENTATION (RAD)
02800          * C
02800   0014          INCLUDE 'MEASOUT.COM'
00100   0015   *      COMMON /MEASOUT/ MX(6),RGPS(6,6),DHCS(2),DVCS(2),MS(3,2),
00200          *     .        RS(2,2,2),DHCL,DVCL,LMU(3),RL(2,2),EMXG(6),
00300          *     .        EDHS(2),EDVS(2),EDHL,EDVL
00400   0016   *      REAL*8 MX,RGPS,DHCS,DVCS,MS,RS,DHCL,DVCL,LMU,RL,EMXG,
00500          *     .        EDHS,_DVS,EDHL,EDVL
00600          * C
00700          * C
00800          * C    MEASUREMENT OUTPUT PARAMETERS
00900          * C                MX       = POSITION/VELOCITY STATE MEASUREMENT - GPS
01000          * C                             (KM,KM/SEC)
01100          * C                EMXG     = ESTIMATED POSITION/VELOCITY STATE
01200          * C                             MEASUREMENT - HGPS
01300          * C                RGPS     = STATE MEASUREMENT NOISE COVARIANCE
01400          * C                             (KNOWLEDGE) - GPS (KM**2,KM/SEC**2)
01500          * C                DHCS     = STAR MEASUREMENT HORIZONTAL DEVIATION
01600          * C                             FROM BORESIGHT - START (RAD)
01700          * C                DVCS     = STAR MEASUREMENT VERTICAL DEVIATION
01800          * C                             FROM BORESIGHT - START (RAD)
01900          * C                EDHS     = ESTIMATED STAR MEASUREMENT HORIZONTAL
02000          * C                             DEVIATION FROM BORESIGHT (RAD)
02100          * C                EDVS     = ESTIMATED STAR MEASUREMENT VERTICAL
02200          * C                             DEVIATION FROM BORESIGHT (RAD)
02300          * C                MS       = STAR MEASUREMENT UNIT VECTOR (SECOND
02400          * C                             SUBSCRIPT REFERS TO TRACKER) - START
02500          * C                RS       = STAR MEASUREMENT NOISE COVARIANCE
```

```
      * C                                (KNOWLEDGE) - START (RAD**2)
      * C             DHCL     = LANDMARK MEASUREMENT HORIZONTAL DEVIATION
      * C                               FROM BORESIGHT - LAMKT (RAD)
      * C             DVCL     = LANDMARK MEASUREMENT VERTICAL DEVIATION
      * C                               FROM BORESIGHT - LAMKT (RAD)
      * C             EDHL     = ESTIMATED LANDMARK MEASUREMENT HORIZONTAL
      * C                               DEVIATION FROM BORESIGHT (RAD)
      * C             EDVL     = ESTIMATED LANDMARK MEASUREMENT VERTICAL
      * C                               DEVIATION FROM BORESIGHT (RAD)
      * C             LMU      = LANDMARK MEASUREMENT UNIT VECTOR - LAMKT
      * C             RL       = LANDMARK MEASUREMENT NOISE COVARIANCE
      * C                               (KNOWLEDGE) - LAMKT (RAD**2)
      * C
0017              DIMENSION TBI(3,3),TIB(3,3),TBS(3,3),US(3),UB(3),UI(3),
          .                 ETBS(3,3),ETSB(3,3),ETBI(3,3),ETSI(3,3),EUI(3),
          .                 PEUS(3,10),PETBI(3,3),UIP(3),ATEMP(4),T1(3,3),
          .                 T2(3,3),EUS(3),TEMP(3)
0018              REAL*8 EUS,ETBI,ETBS,ETSB,ETSI,EUI,PETBI,PEUS,TBI,TBS,TIB,UB,
          .              UI,UIP,US,ATEMP,T1,T2,TEMP
        C
        C*************************************************************
        C         COMPUTE TIB - TRUE BODY TO INERTIAL TRANSFORMATION
        C*************************************************************
0019              DO 5 I=1,4
0020              ATEMP(I)=STAT (I+6)
0021    5         CONTINUE
0022              CALL AMAT(ATEMP,TBI)
0023              CALL MINV3(TBI,TIB)
        C*************************************************************
        C         COMPUTE TBS - TRUE STAR TRACKER TO BODY TRANSFORMATION
        C*************************************************************
0024              DO 100 I=1,3
0025              DO 100 J=1,3
0026              T1(I,J)=TBNS(I,J,K)
0027    100       T2(I,J)=TNS(I,J,K)
0028              CALL MATAB(T1,T2,TBS,3,3,3)
        C*************************************************************
        C         COMPUTE UNIT VECTOR TO STAR IN TRACKER COORDINATES
        C*************************************************************
0029              US(1)=0.
0030              US(2)=0.
0031              US(3)=1.
        C*************************************************************
        C         TRANSFORM TO INERTIAL SPACE
        C*************************************************************
0032              CALL MATAB(TBS,US,UB,3,3,1)
0033              CALL MATAB(TIB,UB,UI,3,3,1)
        C*************************************************************
        C         COMPUTE ETSB - ESTIMATE OF TSB
        C*************************************************************
0034              DO 110 I=1,3
0035              DO 110 J=1,3
0036              T1(I,J)=TBNS(I,J,K)
0037    110       T2(I,J)=TNSK(I,J,K)
0038              CALL MATAB(T1,T2,ETBS,3,3,3)
0039              CALL MINV3(ETBS,ETSB)
        C*************************************************************
```

```
                C       COMPUTE EUI - ESTIMATED UNIT VECTOR IN STAR TRACKER
                C                       COORDINATES
                C************************************************************
      0040              CALL AMAT(E,ETBI)
      0041              CALL MATAB(ETSB,ETBI,ETSI,3,3,3)
      0042              CALL MATAB(ETSI,UI,EUS,3,3,1)
                C
                C************************************************************
                C       CALCULATE ESTINATED MEASUREMENT
                C************************************************************
                C
      0043              EDVS(K)=ASIN( US(1))
      0044              EDHS(K)=ASIN( US(2)/COS(EDVS(K)))
                C************************************************************
                C       COMPUTE PARTIALS OF UNIT VECTOR
                C************************************************************
      0045              DO 10 I=1,3
      0046              DO 10 J=1,10
      0047      10      PEUS(I,J)=0.
      0048              DO 20 J=7,10
                C************************************************************
                C       COMPUTE PARTIALS OF ETBI WRT Q(J),J=7,10
                C************************************************************
      0049              CALL SPETBI(J,PETBI)
                C************************************************************
                C       SET UP ARRAY FOR PARTIALS OF ESTIMATED UNIT VECTOR
                C************************************************************
      0050              CALL MATAB(PETBI,UI,UIP,3,3,1)
      0051              CALL MATAB(ETSB,UIP,TEMP,3,3,1)
      0052              DO 25 I=1,3
      0053      25      PEUS(I,J)=TEMP(I)
                C************************************************************
                C       CONVERT UNIT VECTOR PARTIALS TO MEASUREMENT PARTIALS
                C************************************************************
      0054      20      CALL PMEAS(TEMP,EUS,PDHS(J,K),PDVS(J,K),1)
      0055              DO 30 I=11,22
      0056              PDHS(I,K)=0.
      0057      30      PDVS(I,K)=0.
      0058              DO 40 I=1,6
      0059              PDHS(I,K)=0.
      0060      40      PDVS(I,K)=0.
      0061              RETURN
      0062              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	485	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	8	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	1476	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 PART	2112	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 STARPAR	560	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 ENVIR	404	PIC OVR REL GBL SHR NOEXE RD WRT LONG
6 ASTATE	160	PIC OVR REL GBL SHR NOEXE RD WRT LONG
7 MEASOUT	648	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		HSTAR

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
7-000001E0	R*8	DHCL	7-000001E8	R*8	DVCL	7-00000278	R*8	EDHL	7-00000280	R*8	EDVL
2-00000488	I*4	I	5-00000190	I*4	INIT	2-0000048C	I*4	J	AP-00000004@	I*4	K

A-267

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-000003A8	R*8	ATEMP	32	(4)
4-00000000	R*8	BS	32	(2, 2)
4-00000160	R*8	BSK	32	(2, 2)
6-00000070	R*8	D	24	(3)
6-00000088	R*8	DD	24	(3)
6-00000000	R*8	DE	32	(4)
7-00000150	R*8	DHCS	16	(2)
7-00000160	R*8	DVCS	16	(2)
6-00000020	R*8	E	32	(4)
7-00000258	R*8	EDHS	16	(2)
7-00000268	R*8	EDVS	16	(2)
7-00000228	R*8	EMXG	48	(6)
2-00000180	R*8	ETBI	72	(3, 3)
2-00000120	R*8	ETBS	72	(3, 3)
2-00000168	R*8	ETSB	72	(3, 3)
2-000001F8	R*8	ETSI	72	(3, 3)
2-00000240	R*8	EUI	24	(3)
2-00000458	R*8	EUS	24	(3)
7-000001F0	R*8	LMU	24	(3)
7-00000170	R*8	MS	48	(3, 2)
7-00000000	R*8	MX	48	(6)
3-000006E0	R*8	PDHL	176	(22)
3-00000420	R*8	PDHS	352	(22, 2)
3-00000790	R*8	PDVL	176	(22)

```
3-00000580  R*8  PDVS          352  (22, 2)
2-00000348  R*8  PETBI          72  (3, 3)
2-00000258  R*8  PEUS          240  (3, 10)
5-00000050  R*8  PROFILE       320  (10, 4)
3-00000000  R*8  PX            176  (22)
3-00000210  R*8  PXD           176  (22)
3-00000080  R*8  PY            176  (22)
3-000002C0  R*8  PYD           176  (22)
3-00000160  R*8  PZ            176  (22)
3-00000370  R*8  PZD           176  (22)
7-00000030  R*8  RGPS          288  (6, 6)
7-00000208  R*8  RL             32  (2, 2)
7-000001A0  R*8  RS             64  (2, 2, 2)
6-00000058  R*8  SF             24  (3)
4-00000020  R*8  SS             32  (2, 2)
4-00000180  R*8  SSK            32  (2, 2)
5-00000000  R*8  STATE          80  (10)
2-000003C8  R*8  T1             72  (3, 3)
2-00000410  R*8  T2             72  (3, 3)
2-00000000  R*8  TBI            72  (3, 3)
4-000000D0  R*8  TBNS          144  (3, 3, 2)
2-0000009C  R*8  TBS            72  (3, 3)
2-00000470  R*8  TEMP           24  (3)
2-00000048  R*8  TIB            72  (3, 3)
4-00000040  R*8  TNS           144  (3, 3, 2)
4-000001A0  R*8  TNSK          144  (3, 3, 2)
2-000000F0  R*8  UB             24  (3)
2-00000108  R*8  UI             24  (3)
2-00000390  R*8  UIP            24  (3)
2-000000D8  R*8  US             24  (3)
6-00000040  R*8  WD             24  (3)


LABELS

   Address   Label      Address   Label      Address   Label      Address   Label      Address   Label      Address   Label

     **      5            **      10           **      20           **      25           **      30           **      40
     **      100          **      110


FUNCTIONS AND SUBROUTINES REFERENCED

AMAT       MATAB      MINV3      MTH$DASIN      MTH$DCOS      PMEAS      SPETBI


Total Space Allocated = 5853 Bytes


COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

Run Time:	2.72 seconds
Elapsed Time:	45.11 seconds
Page Faults:	398
Dynamic Memory:	160 pages

2.7.3 Make Measurement Updates (UPDATE)

A Carlson Square Root filter shall be used to estimate the spacecraft attitude and position. The filter calculates a new estimate for the state and model parameters each time a measurement becomes available. The correction depends on the difference between the actual measurement and the predicted measurement and the value of the gain matrix computed from the covariance of the measurement noise and the covariance of the state estimate. Every time the state and model parameters are corrected by the filter, the covariance matrix for the state and model parameters is also changed to reflect the additional knowledge contained in the measurement.

A triangular formulation is used to avoid the possibility of a covariance matrix which is not positive semi-definite and the resulting divergence of the state estimate and failure of the recursive algorithm. The square root formulation provides twice the effective numerical precision of Kalman's original formulation and guarantees a positive definite covariance matrix.

The square root S of the covariance matrix P is defined by

$$P = SS^T$$

The Cholesky method for calculating the square root of a matrix shall be used to calculate the square root S of the covariance matrix P at each measurement update. The equations which define the Cholesky method are

$$S_{ii} = \left[P_{ii} - \sum_{k=i+1}^{n} S_{ik}^2 \right]^{\frac{1}{2}}$$

$$S_{ji} = \left[P_{ji} - \sum_{k=i+1}^{n} S_{jk} S_{ik} \right] \cdot S_{ii}^{-1} \quad , \; j = i-1, 1$$

where

n = dimension of covariance matrix P.
P_{ij} = element of P matrix.
S_{ij} = element of S matrix.

The matrix S is upper triangular. The equation for the elements of S must be evaluated with the initial index $i=n$. The index i is decremented by 1 until the final value $i=1$ is achieved. Evaluation in this order causes no references to uncalculated elements of the S matrix.

The filter equations shall use the difference between the reference measurement and the actual meaurement to update the state measurement. It uses the current covariance matrix which represents the estimate uncertainties in the estimated and considered parameters.

The Carlson formulation does not treat "consider" parameters. Therefore, this formulation has been slightly modified to take non-estimated uncertain model parameters into consideration. Very briefly stated, these modifications simply require that the augmented state covariance (including the measurement consider parameters) replace the spacecraft's position and velocity covariance in Carlson's algorithm.

However, the state and state covariance of the measurement consider parameters are not themselves affected. In this way, the uncertainties in the measurement consider parameters propagate into the estimation of the spacecraft's position vector as if the consider parameters were an extension of the solution vector, but stopping short of actually computing their corrections.

The classic Kalman formulation for a measurement update can be written in three steps

$$P^+ = P - KGP$$

$$X^+ = X + K \cdot dy$$

$$K = PG^T(GPG^T + r)^{-1}$$

where the (+) supercripts denote the state or state covariance following the measurement update,

$P = P'$, the augmented state covariance matrix (algorithm NR-2)

X = State vector (position and velocity, possibly augmented with the moon's mean radius)

K = Kalman gain matrix

$G = G'$, the augmented observation matrix (see algorithm NO-2)

r = Measurement "noise" covariance matrix

dy = Observed minus computed measurement discrepancy vector.

We may express the state covariance matrix as the product of its square-root matrix, S, which has the upper triangular form:

$$P = SS^T$$

With which we may write

$$P^+ = SS^T - SS^T G^T \ (FPG^T + r)^{-1} \ GSS^T$$
$$= S \left[I - (S^T G^T GS)/\alpha\right] S^T$$

Since the Space Sextant measurement is scalar,

$$\alpha = (GPG^T + r)$$

is also scalar, obviating the need for a matrix inversion.

We observe tnat the square-root of P may be updated independently

$$S^+ = S \left[I - ff^T/\alpha\right]^{\frac{1}{2}}$$

where, as in Carlson's notation, we use $f = ST\ GT$

The essence of Carlson's algorithm is the technique for taking the square-root of the bracketed quantity which is done without directy resorting to the Cholesky decomposition.

With each measurement compute the square-root of the augmented state covariance which has been integrated up to the time of the measurement

$$S = \sqrt{P}$$

where S has the upper triangular form by Cholesky's method. Compute the column vector f

$$f = S^T G^T$$

where the segmented G matrix (see algorithm NO-2) is a (1 x n),

n = m (state variables + 1 ("consider" variables)

Compute the diagonal elements of the a^D and C^D matricces:

$$\left.\begin{aligned} \alpha_o &= r \\ \alpha_i &= \alpha_{i-1} + f_i^2 \\ a_i &= (\alpha_{i-1}/d_i)^{\frac{1}{2}} \\ C_i &= f_i/(\alpha_{i-1}\cdot\alpha_i)^{\frac{1}{2}} \end{aligned}\right\} \begin{aligned} &i = 1,n \\ &\text{Note that } \alpha = \alpha_n \end{aligned}$$

and the f* matrix

$$f_1^* = 0$$
$$f_i^* = [\overbrace{f_1 f_2 \cdots f_{i-1}}^{(i-1)} \quad \overbrace{0}^{n-i+1}]^T, \quad i=2,n$$

and form the square-root of the matrix

$$A = a^D - f^* C^D$$

The updated square-root state covariance matrix is thus

$$S(+) = SA$$

from which a fully reconstituted state covariance is computed in temporary storage, including the pseudo-updated considered paramenters,

$$P_o = S^{(+)} S^{(+)T}$$

The actual updated state covariance is constructed from only those elements of P_o which apply to the real estimation variables. The remaining elements of P^+ (1 consider parameters) being unchanged.

$$P_{ij}^{+} = P_{o_{ij}} \} \begin{array}{l} i = 1,m \\ j = 1,n \end{array}$$

$$P_{ij}^{+} = P_{o_{ij}} \} \begin{array}{l} i = m+1,n \\ j = 1,m \end{array}$$

$$P_{ij}^{+} = P_{ij} \} \begin{array}{l} i = m+1,n \\ j = m+1,n \end{array}$$

The state vector itself is updated

where $\quad X^{+} = X + \frac{\overline{b}}{\alpha} dy$

$$b_1 = S_{11} f_1 + S_{12} f_2 \cdots S_{1j} f_j \qquad j = 1,m$$

$$b_2 = S_{22} f_2 + S_{23} f_3 \cdots S_{2j} f_j \qquad j = 1,m$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$b_i = \Sigma S_{ij} f_j \} \begin{array}{l} i = 1,m \\ j = 1,m \end{array}$$

MAKE MEASUREMENT UPDATE VCLR DESCRIPTION

The following three pages constitute the VCLR for the module. The following area and data descriptions will prove helpful for following it.

Area A:

Cholesky Decompostion.
Finds the square root of the covariance matrix and puts it into a one dimensional array (Reference: Carlson, Neal A., "Fast Triangular Formation of the Square Root Filter," AIAA Journal)

Area B:

Upadate parameters with are specified to be estimated. (Reference: Same as for Area A)

Area C:

Put single indexed arrays corresponding to square roots of the covariance into double dimensional arrays.

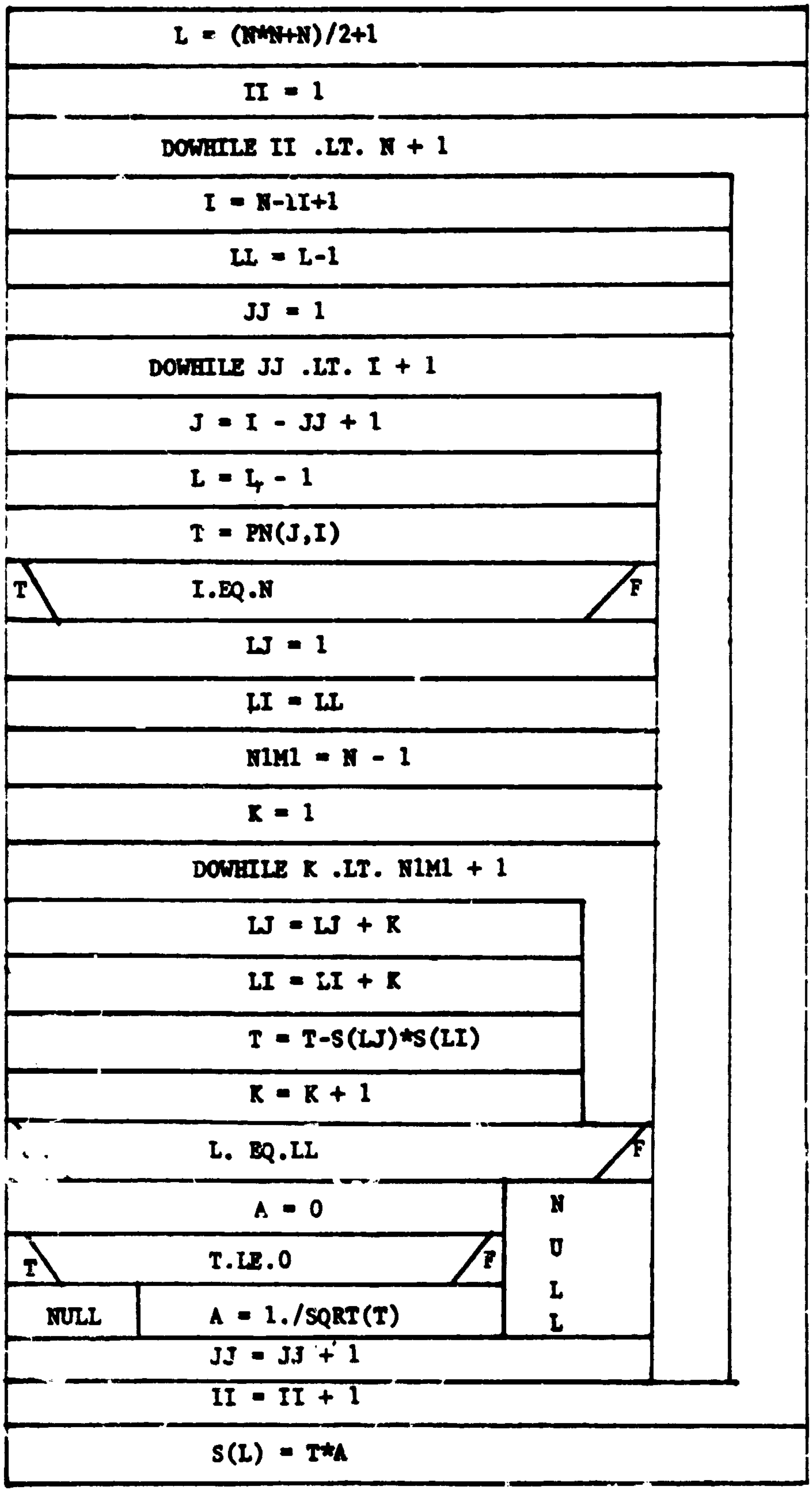
Area D:

Multiply double dimensional arrays to update covariance; however, do not update covariance when neither the row nor the column element was estimated.

Definition of data names necessary for understanding of VCLR.

N = Maximum number of considered parameters
S = Single dimensional square root of the covariance matrix
R = Measurement noise variance
H = Measurement partials array
IPT = Array specifying if parameter is to be estimated
  0= Considered only
  1 = Estimated and considered
X = Array of parameters
PO = Two dimensional square root of covariance matrix
PT = PO transpose
PN = New covariance matrix
DZ = Actual-predicted measurement

UPDATE VCLR (continued)



AREA A

Figure A-43

UPDATE VCLR (continued)

```
L = 0
AL = R
I = 1
DOWHILE I .LT. N + 1
    T = 0.
    K = 1
    DOWHILE K .LT. I + 1
        L = L + 1
        T = T + S(L)*H(K)
        F(I) = T
        K = K + 1
    I = I + 1
L = 0
I = 1
DOWHILE I .LT. N + 1
    B(I) = 0.
    BT = AL
    AL = AL+F(I)**2
    GM = SQRT(AL*BT)
    A = BT/GM
    C = F(I)/GM
    J = 1
    DOWHILE J .LT. I + 1
        L = L + 1
        T = S(L)
        S(L) = A*T-C*B(J)
        B(J) = B(J) + T*F(I)
        J = J + 1
    I = I + 1
I = 1
DOWHILE I .LT. N + 1
    IPT(I) .EQ. 0
      T: NULL
      F: X(I) = X(I) + B(I)*DZ/AL
    I = I + 1
```

AREA

B

Figure A-43

UPDATE VCLR (continued)

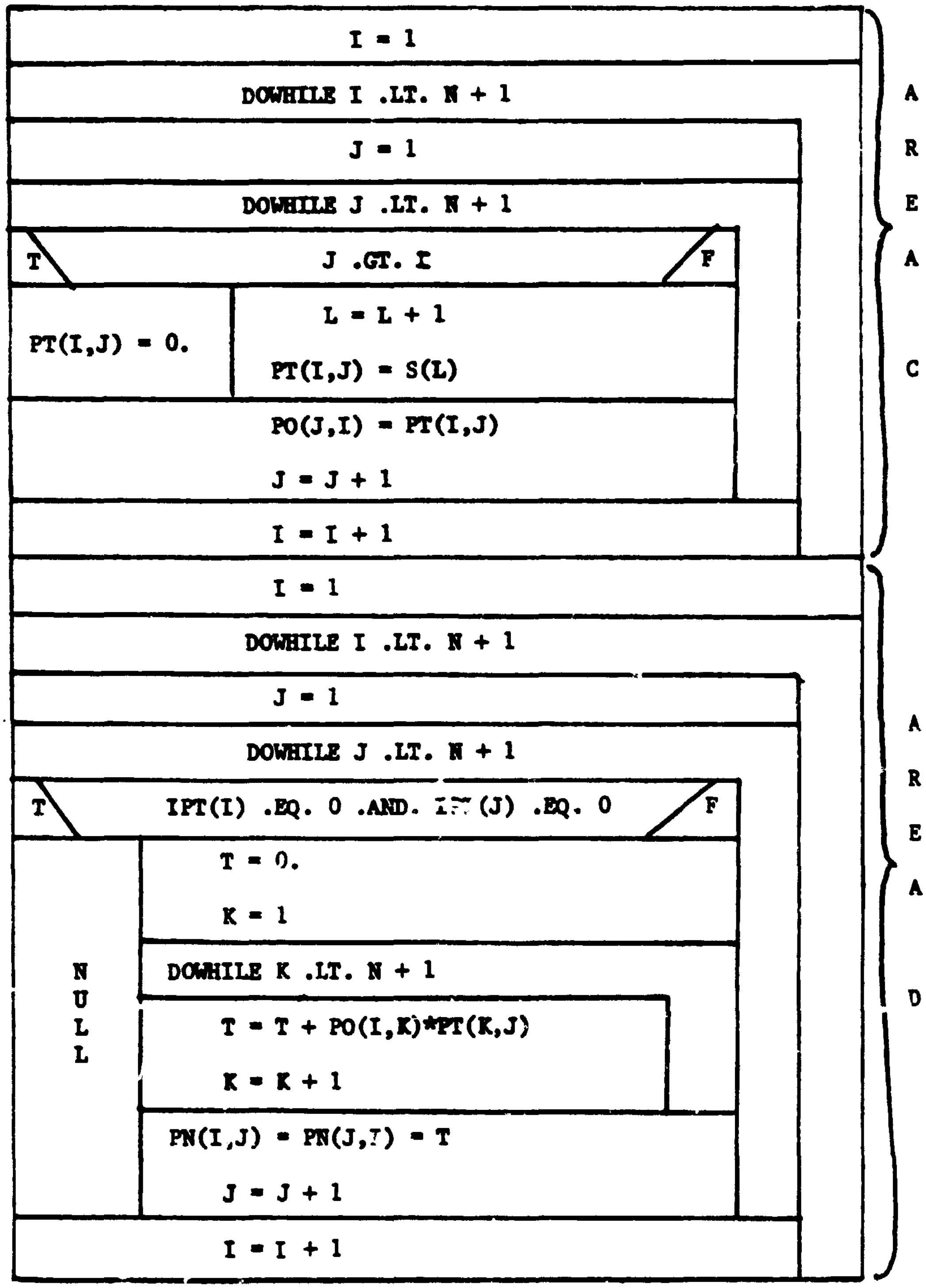


Figure A-43

```
0001          SUBROUTINE UPDATE(N,NS,X,PN,PO,H,R,DZ,IPT)
         C     DIMENSION X(N),PN(N,N),PO(N,N),H(N),IPT(N)
0002          DIMENSION X(22),PN(22,22),PO(22,22),H(22),IPT(22)
0003          DIMENSION  S(253),PT(22,22),F(22),B(22)
0004          REAL*8 A,AL,BT,C,CORREC,DZ,GM,R,T,B,F,H,PN,PO,PT,S,X
0005          INCLUDE 'GAIN.COM'
       * C
       * C
0006   *        COMMON /GAIN/GAIN(22)
0007   *        REAL*8 GAIN
       * C
       * C
         C*********************************************************************
         C        THE FORMAL PARAMETERS ARE;
         C                 N       LARGEST NUMBER OF CONSIDERED PARAMETERS
         C                 NS      NUMBER OF STATE PARAMETERS
         C                 X       ARRAYS OF ESTIMATED PARAMETERS
         C                 PN      NEW COVARIANCE MATRIX
         C                 PO      OLD COVARIANCE MATRIX
         C                 H       MEASUREMANT PARTIALS ARRAYS
         C                 R       MEASUREMENT NOISE VARIANCE
         C                 DZ      ACTUAL - PREDICTED MEASUREMENT DIFFERENCE
         C                 FHIP    STATE TRANSITION MATRIX
         C                 IEST    ARRAYS SPECIFYING IF A PARAMETER IS TO BE
         C                         ESTIMATED OR CONSIDERED ONLY
         C
         C
         C                 NOTE: PO AND PN ENTER WITH THE SAME VALUES AND PO
         C                         IS DISTROYED BY INTERNAL PROCESSING.
         C
         C*********************************************************************
0008          INCLUDE 'DEBUG.COM'
0009   *      COMMON /DEBUG/ IENTER,IDEBUG
       * C
       * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
       * C
       * C                     IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
       * C                     IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
       * C
0010          INCLUDE 'TIME.COM'
       * C
0011   *      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
       *     .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0012   *      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
       *     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
       * C
       * C              THESE ARE THE TIME REFERENCE FRAMES
       * C
       * C                     TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
       * C                     TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
       * C                     TSTOP      RUN TERMINATION TIME (SEC)
       * C                     TIA        ATTITUDE INTEGRATION TIME (SEC)
       * C                     DEL           "          "      STEP SIZE (SEC)
       * C                     TIN        POSITION INTEGRATION TIME (SEC)
       * C                     DTN           "          "      STEP SIZE (SEC)
       * C                     DATEO      DATE OF FLIGHT EPOCH (JD)
       * C                     DATER      DATE OF 1950 EPOCH (JD)
```

```
* C                     TZERO     START TIME IN SECS. SINCE DATE0
* C                     TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
* C                     TIS       REAL WORLD REFERENCE TIME (SEC)
* C                     TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
* C                     DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
* C                               TOO LARGE AT MEASUREMENT TIME
* C                     TPRINT    TIME FOR PRINT (SEC)
* C                     DTPRINT   INCREMENT ON TPRINT (SEC)
  C***********************************************************************
  C
  C                       CHOLESKY DECOMPOSITION    S = SQRT(P)
  C
  C                       CARLSON SQRT FORMULATION
  C                          P = SST
  C
  C                       CHOLESKY DECOMPOSITION
  C                       S = SQUARE ROOT OF P
  C***********************************************************************
        IF(IENTER .EQ. 1) WRITE(6,10)
   10   FORMAT('    ENTERING UPDATE     ')
        IF(IDEBUG .GT. 1) WRITE(6,20)
   20   FORMAT('      INPUT DATA      ')
        IF(IDEBUG .GT. 1) WRITE(6,21) N,NS,R,DZ
   21   FORMAT('    N= ',I3,'    NS= ',I3,'     R= ',E20.10,
     *  '     DZ= ',E20.10)
        IF(IDEBUG .GT. 1) WRITE(6,122)
  122   FORMAT('       X')
        IF(IDEBUG .GT. 1) WRITE(6,22) (X(I),I=1,10)
        IF(IDEBUG .GT. 1) WRITE(6,123)
  123   FORMAT('       PN')
        IF(IDEBUG .GT. 1) WRITE(6,22)((PN(I,J),J=1,10),I=1,10)
        IF(IDEBUG .GT. 1) WRITE(6,124)
  124   FORMAT('       PO')
        IF(IDEBUG .GT. 1) WRITE(6,22)((PO(I,J),J=1,10),I=1,10)
        IF(IDEBUG .GT. 1) WRITE(6,125)
  125   FORMAT('       H')
        IF(IDEBUG .GT. 1) WRITE(6,22)(H(I),I=1,10)
   22   FORMAT(2X,5E20.10,/,7X,5E20.10)
      L = (N*N+N)/2+1
      DO 80 II=1,N
      I = N-II+1
      LL = L-1
      DO 80 JJ=1,I
      J = I-JJ+1
      L = L-1
      T = PN(J,I)
      IF(I.EQ.N) GO TO 70
      LJ = L
      LI = LL
      N1M1 = N-1
      DO 60 K=I,N1M1
      LJ = LJ+K
      LI = LI+K
   60 T = T-S(LJ)*S(LI)
   70 IF(L.NE.LL) GO TO 80
      A = 0.
      IF(T.LE.0.) GO TO 80
```

```
              A = 1./SQRT(T)
       80     S(L) = T*A
       C       COVARIANCE AND STATE UPDATE
       C       S+ = SA
              L = 0
              AL = R
              DO 100 I=1,N
              T = 0.
              DO 90 K=1,I
              L = L+1
       90     T = T+S(L)*H(K)
       100    F(I) = T
              L = 0
              DO 110 I=1,N
              B(I) = 0.
              BT = AL
              AL = AL+F(I)**2
              GM = SQRT(AL*BT)
              A = BT/GM
              C = F(I)/GM
              DO 110 J=1,I
              L = L+1
              T = S(L)
              S(L) = A*T-C*B(J)
       110    B(J) = B(J)+T*F(I)
              DO 120 I=1,N
              IF(IPT(I).EQ.0) GO TO 120
              CORREC = B(I)*DZ/AL
                GAIN(I)=B(I)/AL           ! TEMPORARY TO LOOK AT GAIN - JACK
              X(I) = X(I) + CORREC
              IF(IDEBUG.GT.1) WRITE(6,115) I,GAIN(I),CORREC
       115    FORMAT('     UPDATE GAIN AND CORRECTION FOR ELEMENT ',I2,' = ',
             .          2E20.10)
       120    CONTINUE
       C*********************************************************************
       C         TRANSFORM UPPER TRIANGULAR COVARIANCE MATRIX (S) INTO
       C                 CONVENTIONAL SQUARE MATRIX (PN)
       C*********************************************************************
              L = 0
       C*********************************************************************
       C                   PUT SINGLE ARRAY S INTO DOUBLE ARRAYS PO AND PT WHERE
       C                         PT IS THE TRANSPOSE
       C*********************************************************************
              DO 160 I=1,N
              DO 160 J=1,N
              IF(J.GT.I) GO TO 150
              L = L+1
              PT(I,J) = S(L)
              GO TO 160
       150    PT(I,J) = 0.
       160    PO(J,I) = PT(I,J)
       C*********************************************************************
       C                   MULTIPLY SST TO GET PN, DO NOT UPDATE THE COVARIANCE
       C                         WHERE THE ROWS AND COLUMNS CROSS
       C*********************************************************************
              DO 190 I=1,N
              DO 180 J=I,N
```

```
      IF(IPT(I).EQ.0.AND.IPT(J).EQ.0) GO TO 190
      T = 0.
      DO 170 K=1,N
170   T = T+PO(I,K)*PT(K,J)
      PN(J,I) = T
180   PN(I,J) = T
190   CONTINUE
      RETURN
      END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	1447	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1 $PDATA	224	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2 $LOCAL	6456	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 GAIN	176	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5 TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		UPDATE

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00001868	R*8	A	2-00001870	R*8	AL	2-00001878	R*8	BT	2-00001880	R*8	C
2-00001888	R*8	CORREC	5-00000038	R*8	DATEO	5-00000070	R*8	DATER	5-00000020	R*8	DEL
5-00000068	R*8	DTA	5-00000030	R*8	DTN	5-00000080	R*8	DTPRINT	AP-00000020@	R*8	DZ
2-00001890	R*8	GM	2-000018A0	I*4	I	4-00000004	I*4	IDEBUG	4-00000000	I*4	IENTER
2-000018AC	I*4	II	2-000018A4	I*4	J	2-000018B4	I*4	JJ	2-000018C4	I*4	K
2-000018A8	I*4	L	2-000018BC	I*4	LI	2-000018B8	I*4	LJ	2-000018B0	I*4	LL
AP-00000004@	I*4	N	2-000018C0	I*4	N1M1	AP-00000008@	I*4	NS	AP-0000001C@	R*8	R
2-00001898	R*8	T	5-00000018	R*8	TIA	5-00000000	R*8	TIME	5-00000028	R*8	TIN
5-00000058	R*8	TIS	5-00000060	R*8	TISN	5-00000048	R*8	TMEAS	5-00000008	R*8	TNEXT
5-00000078	R*8	TPRINT	5-00000050	R*8	TRACK	5-00000010	R*8	TSTOP	5-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00001788	R*8	B	176	(22)
2-00001708	R*8	F	176	(22)
3-00000000	R*8	GAIN	176	(22)
AP-00000018@	R*8	H	176	(22)
AP-00000024@	I*4	IPT	88	(22)

```
AP-0000010@ R*8  PN     3872  (22, 22)
AP-0000014@ R*8  PO     3872  (22, 22)
 2-000007E8  R*8  PT     3872  (22, 22)
 2-00000000  R*8  S      2024  (253)
AP-000000C@ R*8  X       176  (22)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000000	10'	1-0000001A	20'	1-00000035	21'	1-00000095	22'	**	60	0-000002E4	70
0-00000300	80	**	90	**	100	**	110	1-000000A5	115'	0-00000490	120
1-0000006A	122'	1-00000075	123'	1-00000080	124'	1-0000008B	125'	0-000004D6	150	0-000004E3	160
**	170	**	180	0-0000059D	190						

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DSQRT

Total Space Allocated = 8447 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 4.71 seconds
Elapsed Time: 86.74 seconds
Page Faults: 386
Dynamic Memory: 160 pages

2.8.0 PRINT PERFORMACE DATA (OUTDATA)

The purpose of this module is to prepare and print output data pertaining to each attitude and navigation time and measurement update.

The Print Performance Module is straightforward as described in the VCLR shown in Figure A-44 . However, one portion which needs explanation, is the section involving the T44 transformation matrix. This is explained below.

The error in the quaternion is found through the equation:

$$\varepsilon = R \cdot F^*$$

where

ε = error in the quaternion

R = real world quaternion

F = filter (or predicted) quaternion

This corresponds intuitively to "subtracting" the predicted values of the quaternion from the real world values.[1]

The covariance matrix associated with the error quaternion is calculated by

$$P_\varepsilon = \left[T_{44}\right] \left[P_F\right] \left[T_{44}\right]^T$$

where P_ε =uncertainty in rotation about each axis

T44 = transition matrix

P_F = filter covariance

The T44 matrix is found by taking the partials of ε with respect to each component of the real world quaternion.

---

1 An alternate equation, $\varepsilon = F^* R$ could have been used. However, this equation is slightly less intuitive and was not chosen for this reason. It should be noted that these two equations yield slightly different results due to the fact that quaternion multiplication is not commutitive.

$$
\epsilon = \begin{aligned}
&\left(F_1R_1 + F_2R_2 + F_3R_3 + F_4R_4\right) \\
+i &\left(-F_2R_1 + F_1R_2 - F_4R_3 + F_3R_4\right) \\
+j &\left(-F_3R_1 + F_4R_2 + F_1R_3 - F_2R_4\right) \\
+k &\left(-F_4R_1 - F_3R_2 + F_2R_3 + F_1R_4\right)
\end{aligned}
$$

$$
T44 = \frac{\partial \epsilon}{\partial R} = \begin{bmatrix}
\frac{\partial \epsilon_0}{\partial R_1} & \frac{\partial \epsilon_0}{\partial R_2} & \frac{\partial \epsilon_0}{\partial R_3} & \frac{\partial \epsilon_0}{\partial R_4} \\
\frac{\partial \epsilon_1}{\partial R_1} & \frac{\partial \epsilon_1}{\partial R_2} & \frac{\partial \epsilon_1}{\partial R_3} & \frac{\partial \epsilon_1}{\partial R_4} \\
\frac{\partial \epsilon_2}{\partial R_1} & \frac{\partial \epsilon_2}{\partial R_2} & \frac{\partial \epsilon_2}{\partial R_3} & \frac{\partial \epsilon_2}{\partial R_4} \\
\frac{\partial \epsilon_3}{\partial R_1} & \frac{\partial \epsilon_3}{\partial R_2} & \frac{\partial \epsilon_3}{\partial R_3} & \frac{\partial \epsilon_3}{\partial R_4}
\end{bmatrix}
$$

$$
= \begin{bmatrix}
F_1 & F_2 & F_3 & F_4 \\
-F_2 & F_1 & -F_4 & F_3 \\
-F_3 & F_4 & F_1 & -F_2 \\
-F_4 & -F_3 & F_2 & F_1
\end{bmatrix}
$$

It should be noted that

$$\varepsilon = R \cdot F^{*} = T_{44} \quad R$$

Therefore, T44 can be used both to find the uncertainty in the rotation about each axis (P_{ε}) and the error in the actual vs predicted value of the quaternion.

Next, the uncertainty in the error quaternion ε is transformed to an uncertainty about each axis as follows:

$$P = \begin{bmatrix} 0 & 0 & 0 & 2 \\ 0 & 0 & 2 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix} \quad P_{\varepsilon} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 2 \\ 2 & 0 & 0 \end{bmatrix}$$

$$= 4 \cdot \begin{bmatrix} P_{\varepsilon 44} & P_{\varepsilon 43} & P_{\varepsilon 42} \\ P_{\varepsilon 34} & P_{\varepsilon 33} & P_{\varepsilon 32} \\ P_{\varepsilon 24} & P_{\varepsilon 23} & P_{\varepsilon 22} \end{bmatrix}$$

The standard deviations of the uncertainty about each axis are the square roots of the appropriate diagonal elements of the P matrix. The RSS of these standard deviations is both printed and plotted.

OUTDATA VCLR

T POST-FLIGHT OPERATION F

INITIALIZE REAL WORLD QUATERNIANS

DETERMINE REAL WORLD POSITION AND VELOCITY

NULL

T TIME FOR PRINT F

INCREMENT PRINT TIME

PRINT HEADER, MEASUREMENT TYPE, ACTUAL AND FILTER STATES

COMPUTE T_{44}

COMPUTE ERROR STATE

PRINT ERROR STATE

T ICALL = 0 F

PRINT EARTH, MOON, AND STAR VECTORS

T MEASUREMENT FOR ATTITUDE REFERENCE F

NULL

COMPUTE AND PRINT NAV. STANDARD DEV.

NULL

COMPUTE AND PRINT ATTITUDE STANDARD DEVIATION

PRINT STATE COVARIANCE

PRINT ERROR PARAMETERS

PRINT STATE TRANSITION MATRIX

NULL

WRITE PLOT FILE

Figure A-44

```
0001          SUBROUTINE OUTDATA(ICALL)
            C
            C FORMAL PARAMETERS
            C      ICALL = OUTPUT PRINT INDICATOR
            C              0 INDICATES CALLED AFTER A MEASUREMENT
            C              1 INDICATES CALLED AFTER PROPAGATION
            C
0002          INCLUDE 'DEBUG.COM'
0003      *   COMMON /DEBUG/ IENTER,IDEBUG
          * C
          * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
          * C
          * C                    IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
          * C                    IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
          * C
0004          INCLUDE 'RESIDUALS.COM'
          * C***************************************************************
          * C                                                              *
          * C--- RESIDUALS.COM CONTAINS THE RESIDUALS VALUES FOR THE GPS,  *
          * C--- LANDMARK TRACKER, AND STAR TRACKER MEASUREMENTS           *
          * C                                                              *
          * C***************************************************************
          * C
0005      *      COMMON /RESIDUALS/DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,
          *     *                   DXMGPS(6)
0006      *      REAL*8 DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,DXMGPS
          * C
0007          INCLUDE 'CONTRL.COM'
0008      *   COMMON /CONTRL/ MOP,TINT
0009      *   REAL*8 TINT
          * C
          * C             PROGRAM CONTROL DESCRIPTORS FOR MULTIPLE RUNS
          * C
          * C                    MOP        MODE OF OPERATION
          * C                               1 = PREFLIGHT SIMULATION
          * C                               2 = POSTFLIGHT SIMULATION
          * C                               3 = MONTE CARLO SIMULATION
          * C             TINT       NUMBER OF SECONDS OF FULL OPERATION PER CYCLE
          * C
0010          INCLUDE 'RVEC.COM'
0011      *   COMMON /RVEC/ R(3),RM(3),RO(3),RSM(3),RSO(3),RSS(3),SB(3)
          *  *              ,RA,R2,R3,RSMA,RTG(3)
0012      *   REAL*8 R,RM,RO,RSM,RSO,RSS,SB,RA,R2,R3,RSMA,RTG
          * C
          * C             THESE ARE RADIUS VECTORS IN ECI AND BODY COORDINATES
          * C
          * C                    R          EARTH CENTER TO S/C -   ECI (KM)
          * C                    RM           "      "     MOON -   ECI (KM)
          * C                    RO           "      "     SUN -    ECI (KM)
          * C                    RSM        SPACECRAFT TO MOON -       ECI (KM)
          * C                    RSO             "         SUN -         ECI (KM)
          * C                    RSS        EARTH CENTER TO STAR -  ECI
          * C                    RA         ABSOLUTE OF VECTOR R (KM)
          * C                    R2         SQUARE OF RA (KM 2)
          * C                    R3         CUBE OF RA (KM 3)
          * C                    RSMA       ABSOLUTE OF RSM (KM)
          * C
```

```
      INCLUDE 'ENVIR.COM'
      COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
      REAL*8 STATE,PROFILE
C
C              REAL WORLD STATE PARAMETERS
C
C                   STATE      STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
C                   PROFILE    ATTITUDE PROFILE-TIME (SEC) VS
C                              INERTIAL ANGULAR RATES (RAD/SEC)
C                   INIT       INTEGRATION INITIALIZATION KEY (-1)
C
      INCLUDE 'ASTATE.COM'
C
      COMMON /ASTATE/ DE(4),E(4),WD(3),SF(3),D(3),DD(3)
      REAL*8 DE,E,WD,SF,D,DD
C
C              ATTITUDE STATE AND CONSIDERED PARAMETERS
C
C                   DE         DIFERENTIAL OF QUATERNIONS
C                   E          QUATERNIONS
C                   WD         GYRO DRIFT RATE (RAD/SEC)
C                   SF         GYRO SCALE FACTOR
C                   D          GYRO NON-ORTHOGANALITY (RAD)
C                   DD         GYPO RELATIVE ORIENTATION (RAD)
C
      INCLUDE 'TARGETS.COM'
      COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
      LOGICAL JFLAG
      REAL*8 PI,TPI
C
C              MEASUREMENT SPECIFICATIONS
C
C                   MTYPE      MEASUREMENT TYPE
C                   JFLAG      SET FOR STAR OBSTRUCTION
C                   MCODE        "   "   MEASUREMENT PROCESSING
C                   PI         PI
C                   TPI        2*PI
C
      INCLUDE 'NSTATE.COM'
C
      COMMON /NSTATE/ XD(6),X(6),RADM,RADE
      REAL*8 XD,X,RADM,RADE
C
C              POSITION STATE AND CONSIDERED PARAMETERS
C
C                   XD         STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
C                   X          STATE POSITION PARAMETERS (KM AND KM/SEC)
C                   RADM       RADIUS OF THE MOON (KM)
C                   RADE       EARTH DETECTABLE RADIUS (KM)
C
      INCLUDE 'PHIA.COM'
      COMMON /PHIA/ PA(4,4),TA(4,12),PDA(4,16),PHIA(16,16),
     *         COVA(16,16),POA(16,16),QMAX
      REAL*8 PA,TA,PDA,PHIA,COVA,POA,QMAX
C
C              THESE ARE THE ATTITUDE TRANSITION AND COVARIANCE MATRICES
```

```
C
C                      PA        ATTITUDE STATE TRANSITION MATRIX
C                      TA           "      PARAMETER TRANSITION MATRIX
C                      PDA          "      DERIVATIVE OF TRANSITION MATRICES
C                      PHIA         "      AGGREGATE TRANSITION MATRIX
C                      COVA         "      NEW COVARIANCE MATRIX
C                      POA          "      PREVIOUS COVARIANCE MATRIX
C                      QMAX         "      COVARIANCE NORM MAX
        INCLUDE 'PHIN.COM'
        COMMON /PHIN/ PN(6,6),PDN(6,6),PHIN(6,6),COVN(6,6),
     .          PON(6,6)
        REAL*8 PN,PDN,PHIN,COVN,PON
C
C               THESE ARE THE NAVIGATION TRANSITION AND COVARIANCE ARRAYS
C
C                      PN        POSITION STATE TRANSITION MATRIX
C                      PDN          "      DERIVATIVE OF TRANSITION MATRIX
C                      PHIN         "      AGGREGATE TRANSITION MATRIX
C                      COVN         "      NEW COVARIANCE MATRIX
C                      PON          "      PREVIOUS COVARIANCE MATRIX
C
        INCLUDE 'ARRAYS.COM'
C
        COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
     .          ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
        REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
C
C               THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
C
C                      T1 - T4      SINGLE DIMENSION ARRAYS
C                      T11 - T77    DUAL DIMENSIONED ARRAYS
C                      T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
C
        INCLUDE 'TIME.COM'
C
        COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
     .          ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
        REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
C
C               THESE ARE THE TIME REFERENCE FRAMES
C
C                      TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
C                      TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
C                      TSTOP     RUN TERMINATION TIME (SEC)
C                      TIA       ATTITUDE INTEGRATION TIME (SEC)
C                      DEL          "          "      STEP SIZE (SEC)
C                      TIN       POSITION INTEGRATION TIME (SEC)
C                      DTN          "          "      STEP SIZE (SEC)
C                      DATEO     DATE OF FLIGHT EPOCH (JD)
C                      DATER     DATE OF 1950 EPOCH (JD)
C                      TZERO     START TIME IN SECS. SINCE DATEO
C                      TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
C                      TIS       REAL WORLD REFERENCE TIME (SEC)
C                      TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
C                      DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
```

C-7

```
C                                   TOO LARGE AT MEASUREMENT TIME
C                        TPRINT     TIME FOR PRINT (SEC)
C                        DTPRINT    INCREMENT ON TPRINT (SEC)
C
      INCLUDE 'UPDT.COM'
C
      COMMON /UPDT/ QN(6),QA(16),Q(6,6),QDOT(6,6)
      REAL*8 QN,QA,Q,QDOT
C
C              STATE  STIMATION PARAMETERS
C
C                 QN        NAV. DYN. NOISE COVARIANCE DIAGONAL
C                 QA        MIN. VALUES FOR ATT. COVARIANCE DIAGONAL
C                 Q         CONTRIBUTION TO NAV. COV. FOR DYN. NOISE
C                 QDOT      DIFFERENTIAL OF Q
C
      INCLUDE 'PLOT.COM'
C
      COMMON /PLOT/ TP1,TP2
      REAL*8 TP1,TP2
C
C              PLOTING INFORMATION
C
C                 TP1       LOWER ASSCISSA VALUE - TIME (MIN)
C                 TP2       UPPER    "      "       "
C
      INCLUDE 'COMPOSIT.COM'
      COMMON /COMPOSIT/ PHI(22,22),QT(22,22),COV(22,22),PO(22,22),
     *        IP(22),XT(22),P(22,22)
      REAL*8 PHI,QT,COV,PO,XT,P
C
C             PHI       = COMPOSIT STATE TRANSITION MATRIX
C             QT        =    "      PROCESS NOISE ARRAY
C             COV       = NEWEST COVARIANCE ARRAY
C             PO        = OLD COVARIANCE ARRAY
C             IP        = ARRAY OF FLAGS INDICATING ESTIMATED AND
C                              CONSIDERED PARAMETERS
C             XT        = COMPOSIT ESTIMATED PLUS CONSIDERED
C                              STATE VECTOR
C             P         = INITIALIZED TRANSITION MATRIX FOR NEXT
C                              INTREVAL
C
      INCLUDE 'TITL..COM'
      COMMON /TITLE/ ATITLE(40)
      LOGICAL*1 ATITLE
C
C             ATITL. IS THE TITLE PRINTED AT EACH PRINT TIME
C             AS WELL AS THE TOP OF EACH PLOT
C
      INCLUDE 'LMTPAR.COM'
      COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
     *        BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
      REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
     *        THET
C
C
C     LANDMARK TRACKER PARAMETERS
```

```
          * C             AL      = ALTITUDE OF LANDMARK (KM)
          * C             LON     = LONGITUDE OF LANDMARK (DEG)
          * C             LAT     = LATITUDE OF LANDMARK (DEG)
          * C             TBNL    = ORIENTATION ARRAY FOR LANDMARK TRACKER
          * C                               NOMINAL TO BODY
          * C             TNL     = MISALIGNMENT ARRAY - ACTUAL
          * C                               TRACKER TO NOMINAL
          * C             BL      = BIAS - ACTUAL (RAD)
          * C             SL      = NOISE STANDARD DEVIATION - ACTUAL (RAD)
          * C             BKL     = BIAS - KNOWLEDGE (RAD)
          * C             THET    = LOOK ANGLE (RAD)
          * C             SKL     = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
          * C             TIEO    = INITIAL EARTH FIXED TO INERTIAL
          * C                               TRANSFORMATION
          * C             TNLK    = MISALIGNMENT ARRAY KNOWLEDGE
          * C                               TRACKER TO NOMINAL
          * C             SIGGCP  = POSITION UNCERTAINTY DUE TO CLOUDS
          * C
            C *** ICALL = 1      TIME UPDATE
            C *** ICALL = 0      MEASUREMENT UPDATE
            C
0053              DIMENSION T44T(4,4),PATT(4,4),PAT(4,4)
0054              DIMENSION AT(3,3),ST(4)
0055              DIMENSION DSTAT (10)
0056              REAL*8 DELTAT,DZ,EATT,PATTF,PNAVF,PTIME,TMLAST,TPUNCH,AT,DSTATE,
                 .        PAT,PATT,ST,T44T,ENAV
0057              DATA TMLAST /0./
            C*********************************************************************
            C CALL AEROSPACE ROUTINE TO GET REAL WORLD ATTITUDE (TEMPORARY)
            C*********************************************************************
0058              IF (MOP.NE.2) GO TO 10
0059              STATE(7) = ST(4)
0060              STATE(8) = ST(1)
0061              STATE(9) = ST(2)
0062              STATE(10) = ST(3)
0063              IF (TIME.GT..00001) GO TO 13
0064              INIT1=-1
0065              GO TO 10
            C*********************************************************************
            C DETERMINE REAL WORLD POSITION AND VELOCITY
            C*********************************************************************
0066        13    DELTAT = TIME - TMLAST
0067              IF (DELTAT.LT..00001) GO TO 10
0068              CALL RUNG(INIT1,STATE,DSTATE,TMLAST,DELTAT)
0069              TMLAST = TIME
0070        10      IF(TIME .LT. TPRINT) GO TO 220
0071                WRITE(6,5)(ATITLE(I),I=1,40)
0072        5       FORMAT(1H1,//,4X,40A1)
0073              WRITE(6,14) TIM
0074        14    FORMAT(//,5X,7HTIME = ,F9.3)
0075              GO TO (150,160,170,180),MCODE
0076              WRITE(6,145)
0077              GO TO 15
0078        145   FORMAT(/,5X,14HNO MEASUREMENT)
0079        150   WRITE(6,155)
0080                WRITE(6,1155) THET
0081        1155    FORMAT(/,5X,'LOOK ANGLE = ',E15.7)
```

```
             WRITE(6,1156) LON,LAT,AL
 1156        FORMAT(/,5X,'LON,LAT,AL = ',3(3X,E15.7))
          GO TO 15
 155      FORMAT(/,5X,44HLANDMARK TRACKER MEASUREMENTS HAVE BEEN MADE)
 160      WRITE(6,165)
          GO TO 15
 165      FORMAT(/,5X,31HGPS MEASUREMENTS HAVE BEEN MADE)
 170      WRITE(6,175)
          GO TO 15
 175      FORMAT(/,5X,41HSTARTRACKER 1 MEASUREMENTS HAVE BEEN MADE)
 180      WRITE(6,185)
 185      FORMAT(/,5X,41HSTARTRACKER 2 MEASUREMENTS HAVE BEEN MADE)
 15       WRITE(6,18) STATE,X,E
 18       FORMAT(//5X,50HACTUAL/FILTER STATES -- X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
        . /2(2X,3E14.7,7E12.4/))
 220         CONTINUE
          T44(1,1) = STATE(7)
          T44(2,2) = STATE(7)
          T44(3,3) = STATE(7)
          T44(4,4) = STATE(7)
          T44(2,1) = -STATE(8)
          T44(3,4) = -STATE(8)
          T44(1,2) = STATE(8)
          T44(4,3) = STATE(8)
          T44(3,1) = -STATE(9)
          T44(4,2) = -STATE(9)
          T44(1,3) = STATE(9)
          T44(2,4) = STATE(9)
          T44(4,1) = -STATE(10)
          T44(2,3) = -STATE(10)
          T44(1,4) = STATE(10)
          T44(3,2) = STATE(10)
C**********************************************************************
C PERFORM QUATERNION MULTIPLY
C EPS=(FILTERQUATERNION)*(REALWORLDQUATERNIONSTAR)
C**********************************************************************
          CALL MATAB(T44,E,T4(7),4,4,1)
C**********************************************************************
C            MTYPE = 1 FOR NAVIGATION
C                    2 FOR ATTITUDE REFERENCE
C                    3 FOR CALIBRATION
C                    4 FOR CRUISE FLIGHT
C**********************************************************************
          DO 20 I=1,6
 20       T4(I) = STATE(I)-X(I)
            IF(TIME .LT. TPRINT) GO TO 120
            TPRINT=TPRINT+DTPRINT
 30       WRITE(6,40) T4
 40       FORMAT(5X,18HSTATE ERROR VECTOR/2X,3E14.7,7E12.4,/)
C*****************************************************************
C            WRITE OUT DATA FOR PLOTTING ROUTINES
C*****************************************************************
          IF(ICALL.EQ.0) WRITE(6,115) RM, RO, RSS
 115      FORMAT(4X,' EARTH TO MOON ',7X,3E22.14,/,4X,
        . ' EARTH TO SUN  ',7X,3E22.14,/,4X,
        . ' UNIT VECTOR TO STAR  ',3E22.14)
          IF (MTYPE.EQ.2.AND.ICALL.EQ.0) GO TO 70
```

```
0123            PNAVF= SQRT(COV(1,1)+COV(2,2)+COV(3,3))
0124            DO 50 I=1,3
0125      50    T1(I) = SQRT(COV(I,I))
0126            WRITE(6,60) TIN,T1,PNAVF
0127      60    FORMAT(/,5X,29HNAVIGATION STANDARD DEVIATION,F7.0,4E16.6)
0128            ENAV = SQRT(T4(1)**2+T4(2)**2+T4(3)**2)
0129            IF (MTYPE.EQ.1.AND.ICALL.EQ.0) GO TO 120
0130      70    DO 80 I=1,4
0131            DO 80 J=1,4
0132            T44T(I,J) = T44(J,I)
0133      80    PATT(I,J) = COVA(I,J)
0134            CALL MATAB(PATT,T44T,PAT,4,4,4)
0135            CALL MATAB(T44,PAT,PATT,4,4,4)
0136            PATTF = 2.*SQRT(PATT(2,2)+PATT(3,3)+PATT(4,4))
0137            DO 90 I=2,4
0138      90    PATT(I,I) = 2.*SQRT(PATT(I,I))
0139            WRITE(6,100) TIA,(PATT(I,I),I=2,4),PATTF
0140      100   FORMAT(/5X,27HATTITUDE STANDARD DEVIATION,F7.0,4E16.6)
0141            WRITE(6,992)
0142      992   FORMAT(///4X,' STATE COVARIANCE ')
0143            DO 105 I = 1,22
0144               II = I
0145               IF (I.GT.8) II = 8
0146               WRITE(6,103) (COV(I,J),J=1,II)
0147        103    FORMAT(5X,8E15.8)
0148        105    CONTINUE
0149            WRITE(6,106)
0150        106 FORMAT(1H1,/5X,'-----(CONTINUED)')
0151            DO 107 I = 9,22
0152              II=I
0153              IF(I.GT.16) II=16
0154        107    WRITE(6,103) (COV(I,J),J=9,II)
0155              WRITE(6,108)
0156      108     FORMAT(/5X,'-----(CONTINUED)')
0157              DO 109 I=17,22
0158      109       WRITE(6,103)(COV(I,J),J=17,I)
0159            WRITE(6,110) (E(I),I=1,16)
0160      110   FORMAT(//5X,19HATTITUDE PARAMETERS/5X,11HQUATERNIONS,9X,4E16.9/
               .5X,10HGYRO DRIFT,10X,3E16.9/5X,17HGYRO SCALE FACTOR,3X,3E16.9/
               .5X,20HGYRO NON-ORTHOGONAL ,3E16.9/5X,20HRELATIVE ORIENTATION,
               .3E16.9)
0161            EATT = 2.*SQRT(T4(8)**2+T4(9)**2+T4(10)**2)
0162              WRITE(6,111)
0163      111     FORMAT(//5X,' STATE TRANSITION MATRIX ')
0164              DO 112 I = 1,10
0165                II=I
0166                IF(I.GT.8) II=8
0167      112       WRITE(6,103)(PHI(I,J),J=1,II)
0168              WRITE(6,108)
0169              DO 113 I=9,10     !JACK-2/21/81
0170                II=I
0171              IF(I .GT. 10) II=10    !JACK-2/21/81
0172      113       WRITE(6,103) (PHI(I,J),J=9,II)
          C****************************************************************************
          C         CREATE FORMATTED FILE LISTING TIME, MEASUREMENT TYPE JUST TAKEN,
          C         ESTIMATE OF QUATERNION, DIFFERENCE BETWEEN MEASUREMENT AND PREDICTED
          C         MEASUREMENT, ESTIMATE OF POSITION AND VELOCITY AT CERTAIN TIME PTS.
```

```
                C*********************************************************************
0173    120         WRITE(12,45)TIME,T4,(SQRT(COV(J1,J1)),J1=1,6),
                *            (PATT(J1,J1),J1=1,4),
                *            DZHLM,DZVLM,DZHST1,DZVST1,DZHST2,DZVST2,DXMGPS,
                *            (STAT(J1),J1=1,6)
0174    45          FORMAT(39F20.10)
0175               IF(ICALL.EQ.1) GO TO 130
0176               PTIME = TIA
0177               IF (MTYPE.EQ.1) PTIME = TIN
0178               IF(PTIME.LT.TPUNCH) GO TO 130
        C           WRITE(11,125) PTIME,MTYPE,MSET,
        C          . (E(I),I=1,4),DZ,(X(I),I=1,6)
0179    125        FORMAT(1X,F10.1,1X,2(I2,1X),4(F11.9,1X),E11.5,1X,/,
                   . 8X,6(E11.5,1X))
0180    130        RETURN
0181               END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	2002	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	851	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	912	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	RESIDUALS	96	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	CONTRL	12	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	RVEC	224	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	ASTATE	160	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
9	TARGETS	36	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
10	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
11	PHIA	7176	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
12	PHIN	1440	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
13	ARRAYS	1096	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
14	TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
15	UPDT	752	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
16	PLOT	16	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
17	COMPOSIT	19624	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
18	TITLE	40	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
19	LMTPAR	392	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		OUTDATA

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
19-00000000	R*8	AL	14-00000038	R*8	DATEO	14-00000070	R*8	DATER	14-00000020	R*8	DEL
2-00000238	R*8	DELTAT	14-00000068	R*8	DTA	14-00000030	R*8	DTN	14-00000080	R*8	DTPRINT
2-00000240	R*8	DZ	4-00000000	R*8	DZHLM	4-00000010	R*8	DZHST1	4-00000020	R*8	DZHST2
4-00000008	R*8	DZVLM	4-00000018	R*8	DZVST1	4-00000028	R*8	DZVST2	2-00000248	R*8	EATT
2-00000278	R*8	ENAV	2-00000284	I*4	I	AP-00000004@	I*4	ICALL	3-00000004	I*4	IDEBUG
3-00000000	I*4	IENTER	2-0000028C	I*4	II	7-00000190	I*4	INIT	2-00000280	I*4	INIT1
9-00000004	I*4	IS	2-00000288	I*4	J	2-00000290	I*4	J1	9-0000000C	L*4	JFLAG
19-00000010	R*8	LAT	19-00000008	R*8	LON	9-00000010	I*4	MCODE	5-00000000	I*4	MOP
9-00000000	I*4	MTYPE	9-00000008	I*4	NS	2-00000250	R*8	PATTF	9-00000014	R*8	PI
2-00000258	R*8	PNAVF	2-00000260	R*8	PTIME	11-00001C00	R*8	QMAX	6-000000B0	R*8	R2
6-00000088	R*8	R3	6-000000A8	R*8	RA	10-00000068	R*8	RADE	10-00000060	R*8	RADM
6-000000C0	R*8	RSMA	19-00000178	R*8	SIGGCP	19-00000180	R*8	THET	14-00000018	R*8	TIA
14-00000000	R*8	TIME	14-00000028	R*8	TIN	5-00000004	R*8	TINT	14-00000058	R*8	TIS
14-00000060	R*8	TISN	14-00000048	R*8	TMEAS	2-0000026[illegible]	R*8	TMLAST	14-00000008	R*8	TNEXT
16-00000000	R*8	TP1	16-00000008	R*8	TP2	9-0000001C	R*8	TPI	14-00000078	R*8	TPRINT
2-00000270	R*8	TPUNCH	14-00000050	R*8	TRACK	14-00000010	R*8	TSTOP	14-00000040	R*8	TZERO

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000180	R*8	AT	72	(3, 3)
18-00000000	L*1	ATITLE	40	(40)
19-000000C8	R*8	BKL	16	(2)
19-000000A8	R*8	BL	16	(2)
17-00001E40	R*8	COV	3872	(22, 22)
11-00000C00	R*8	COVA	2048	(16, 16)
12-00000360	R*8	COVN	288	(6, 6)
8-00000070	R*8	D	24	(3)
8-00000088	R*8	DD	24	(3)
8-00000000	R*8	DE	32	(4)
2-000001E8	R*8	DSTATE	80	(10)
4-00000030	R*8	DXMGPS	48	(6)
8-00000020	R*8	E	32	(4)
17-00003C80	I*4	IP	88	(22)
17-00003D88	R*8	P	3872	(22, 22)
11-00000000	R*8	PA	128	(4, 4)
2-00000100	R*8	PAT	128	(4, 4)
2-00000080	R*8	PATT	128	(4, 4)
11-00000200	R*8	PDA	512	(4, 16)
12-00000120	R*8	PDN	288	(6, 6)
17-00000000	R*8	PHI	3872	(22, 22)
11-00000400	R*8	PHIA	2048	(16, 16)
12-00000240	R*8	PHIN	288	(6, 6)
12-00000000	R*8	PN	288	(6, 6)
17-00002D60	R*8	PO	3872	(22, 22)
11-00001400	R*8	POA	2048	(16, 16)
12-00000480	R*8	PON	288	(6, 6)
7-00000050	R*8	PROFILE	320	(10, 4)
15-00000080	R*8	Q	288	(6, 6)
15-00000030	R*8	QA	128	(16)
15-000001D0	R*8	QDOT	288	(6, 6)

15-00000000	R*8	QN	48	(6)
17-00000F20	R*8	QT	3872	(22, 22)
6-00000000	R*8	R	24	(3)
6-00000018	R*8	RM	24	(3)
6-00000030	R*8	RO	24	(3)
6-00000048	R*8	RSM	24	(3)
6-00000060	R*8	RSO	24	(3)
6-00000078	R*8	RSS	24	(3)
6-000000C8	R*8	RTG	24	(3)
6-00000090	R*8	SB	24	(3)
8-00000058	R*8	SF	24	(3)
19-000000D8	R*8	SKL	16	(2)
19-000000B8	R*8	SL	16	(2)
2-000001C8	R*8	ST	32	(4)
7-00000000	R*8	STATE	80	(10)
13-00000000	R*8	T1	24	(3)
13-00000098	R*8	T11	72	(3, 3)
13-00000018	R*8	T2	24	(3)
13-00000030	R*8	T3	24	(3)
13-000000E0	R*8	T33	72	(3, 3)
13-00000048	R*8	T4	80	(10)
13-00000128	R*8	T44	128	(4, 4)
2-00000000	R*8	T44T	128	(4, 4)
13-000003E8	R*8	T5	32	(4)
13-00000408	R*8	T6	32	(4)
13-000001A8	R*8	T66	288	(6, 6)
13-00000428	R*8	T7	32	(4)
13-000002C8	R*8	T77	288	(6, 6)
11-00000080	R*8	TA	384	(4, 12)
19-00000018	R*8	TBNL	72	(3, 3)
19-00000130	R*8	TIEO	72	(3, 3)
19-00000060	R*8	TNL	72	(3, 3)
19-000000E8	R*8	TNLK	72	(3, 3)
8-00000040	R*8	WD	24	(3)
10-00000030	R*8	X	48	(6)
10-00000000	R*8	XD	48	(6)
17-00003CD8	R*8	XT	176	(22)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000000	5'	0-00000070	10	0-00000049	13	1-0000000C	14'	0-00000189	15	1-00000119	18'
**	20	**	30	1-00000165	40'	1-00000345	45'	**	50	1-000001E0	60'
0-00000387	70	**	80	**	90	1-0000020B	100'	1-0000024E	103'	**	105
1-00000256	106'	**	107	1-0000026F	108'	**	109	1-00000285	110'	1-00000325	111'
**	112	**	113	1-000001BA	115'	0-000006D8	120	**	125'	0-000007D1	130
1-0000001D	145'	0-000000FB	150	1-00000064	155'	0-00000173	160	1-00000096	165'	0-00000188	170
1-000000BB	175'	0-000001A3	180	1-000000EA	185'	0-000001F0	220	1-00000234	992'	1-00000031	1155'
1-00000047	1156'										

A-296

```
FUNCTIONS AND SUBROUTINES REFERENCED

MATAB          MTH$DSQRT       RUNG

Total Space Allocated = 35489 Bytes

COMMAND QUALIFIERS

  FORTRAN /L OUTDATA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:           7.31 seconds
  Elapsed Time:       24.43 seconds
  Page Faults:        504
  Dynamic Memory:     109 pages
```

2.9 Plotting and Utility Routines

The following routines are included for completeness. They are all routines associated with GCPSIM as plotting and utility routines.

```
0001            SUBROUTINE CLOTBL(X,PCNT)
           C**************************************************************
           C
           C        SUBROUTINE CLOTBL COMPUTES A PERCENT OF CLOUD COVER
           C        BASED ON THE RANDOM NUMBER X.
           C
           C        INPUT PARAMETERS
           C                X        = A RANDOM NUMBER BETWEEN 0-1
           C        OUTPUT PARAMETERS
           C                PCNT     =THE PERCENT CLOUD COVER
           C        CALLED BY MEASURE
           C
           C**************************************************************
           C
0002            INCLUDE 'CLOUD.COM'
0003     *      COMMON /CLOUD/ CLOTBL(12)
0004     *      REAL*8 CLOTBL
         * C                PCNT         THE PERCENTAGE OF CLOUD COVER
         * C
0005            INCLUDE 'MODE.COM'
0006     *      COMMON /MODE/ MODE(10)
         * C
         * C            MODE(1) = LANDMARK TRACKER SWEEP MODE
         * C                          0 = RANDOM
         * C                          1 = FIXED AT INPUT THET
         * C                          2 = NO DEFAULT TO STAR TRACKER
         * C            MODE(2) = CLOUD SELECTION MODE
         * C                          0 = RANDOM CLOUD DENSITIES BASED
         * C                                  ON INPUT TABLES CLOTBL
         * C                          1 = FIXED DENSITY AT NO CLOUDS
         * C                          2 = NO CLOUDS WITH 100% CLOUD
         * C                                  COVER FOR A SPECIFIED
         * C                                  PERIOD (CLOTBL(11,12))
         * C            MODE(3-10) NOT SPECIFIED AT PRESENT
         * C
0007            INCLUDE 'TIME.COM'
         * C
0008     *      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
         *     .              ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0009     *      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
         *     .         TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
         * C
         * C            THESE ARE THE TIME REFERENCE FRAMES
         * C
         * C                TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
         * C                TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
         * C                TSTOP     RUN TERMINATION TIME (SEC)
         * C                TIA       ATTITUDE INTEGRATION TIME (SEC)
         * C                DEL           "           "      STEP SIZE (SEC)
         * C                TIN       POSITION INTEGRATION TIME (SEC)
         * C                DTN           "           "      STEP SIZE (SEC)
         * C                DATEO     DATE OF FLIGHT EPOCH (JD)
         * C                DATER     DATE OF 1950 EPOCH (JD)
         * C                TZERO     START TIME IN SECS. SINCE DATEO
         * C                TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
         * C                TIS       REAL WORLD REFERENCE TIME (SEC)
         * C                TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
```

```
• C                  DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
• C                             TOO LARGE AT MEASUREMENT TIME
• C                  TPRINT     TIME FOR PRINT (SEC)
• C                  DTPRINT    INCREMENT ON TPRINT (SEC)
• C
                 REAL*8 PCNT
                 PCNT=0.
                 GO TO (20,30),MODE(2)
                 I=X*10+1.
                 IF(I .LT. 1) I=1
                 IF(I .GT. 10) GO TO 10
   C
   C*********************************************************************
   C             INTERPOLATE CLOUD TABLE DATA
   C*********************************************************************
   C
                 PCNT=(CLOTBL(I+1)-CLOTBL(I))*(X*10-I)+CLOTBL(I)
                 GO TO 100
   10            PCNT=CLOTBL(I)
                 GO TO 100
   20            PCNT=0.
                 GO TO 100
   30            IF((TIME.GE.CLOTBL(11)).AND.(TIME.LE.CLOTBL(12)))PCNT=100.
   100           RETURN
                 END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	151	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	4	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 CLOUD	96	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4 MODE	40	PIC OVR REL GBL SHR NOEXE RD WRT LONG
5 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		CLDTBL

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
5-00000038	R*8	DATEO	5-00000070	R*8	DATER	5-00000020	R*8	DEL	5-00000068	R*8	DTA
5-00000030	R*8	DTN	5-00000080	R*8	DTPRINT	2-00000000	I*4	I	AP-00000008@	R*8	PCNT
5-00000018	R*8	TIA	5-00000000	R*8	TIME	5-00000028	R*8	TIN	5-00000058	R*8	TIS
5-00000060	R*8	TISN	5-00000048	R*8	TMEAS	5-00000008	R*8	TNEXT	5-00000078	R*8	TPRINT
5-00000050	R*8	TRACK	5-00000010	R*8	TSTOP	5-00000040	R*8	TZERO	AP-00000004@	R*4	X

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-00000000	R*8	CLDTBL	96	(12)
4-00000000	I*4	MODE	40	(10)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
0-0000005D	10	0-00000068	20	0-00000070	30	0-00000096	100

Total Space Allocated = 427 Bytes

COMMAND QUALIFIERS

FORTRAN /L CLDTBL

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 1.19 seconds
Elapsed Time: 57.87 seconds
Page Faults: 141
Dynamic Memory: 30 pages

```
$!COMP.COM
$!
$!      COMP.COM COMPILES ALL THE NECESSARY ROUTINES  OR EXECUTION
$!      OF THE GCP SIMULATION PROGRAM
$!
$FOR/LIST-        !/DEBUG/NOOPTIMIZE-
        GCP,-
                INDATA,-
                        MATAB,-
                OUTDATA,-
                        RUNG,-
                                DNAV,-
                                        EPHEM,-
                                                TRUEA,-
                                        SPRESS,-
                                                OCCULT,-
                                        GPERT,-
!                       MATAB,-
!               DNAV,-
!                       EPHEM,-
!                               TRUEA,-
!                       SPRESS,-
!                               OCCULT,-
!                       GPERT,-
                GCPSEQ,-
                        VISIBLE,-
!                               OCCULT,-
                GENENV,-
                        TREG,-
                        GYROUT,-
                                RATE,-
                                BMAT,-
                                CMAT,-
!                               MATAB,-
                                MINV3,-
                                GAUSS,-
                                KATT,-
!                                       BMAT,-
!                                       CMAT,-
!                                       MATAB,-
                                        QMULT,-
                                        UNIT,-
!                       RUNG,-
!                               DNAV,-
!                                       EPHEM,-
!                                               TRUEA,-
!                                       SPRESS,-
!                                               OCCULT,-
!                                       GPERT,-
                GYRO,-
!                       KATT,-
!                               BMAT,-
!                               CMAT,-
!                               MATAB,-
!                               QMULT,-
!                               UNIT,-
                        PDATT,-
!                               MATAB,-
                        PATT,-
                MEASURE,-
!                       RANDU,-
                        CLDTBL,-
```

```
          RDGCP,-
                    RDDTA,-
                    AMAT,-
                    MATAB,-
                    MINV3,-
                    WET,-
          BVECT,-
                    MATAB,-
                    AMAT,-
          VISIBLE,-
                    OCCULT,-
          LAMKT,-
                    WET,-
                              MATAB,-
                    MATAB,-
                    AMAT,-
                    MINV3,-
                    UNIT,-
                    GAUSS,-
          GPS,-
                    GAUSS,-
          START,-
                    GAUSS,-
INTG,-
          TREG,-
          RKG,-
                    DNAV,-
                              EPHEM,-
                                        TRUEA,-
                              SPRESS,-
                                        OCCULT,-
                              GPERT,-
                    PDNAV,-
                              MATAB,-
EST,-
          PROP,-
          MPART,-
                    HLMT,-
                              WET,-
                                        MATAB,-
                              MATAB,-
                              MINV3,-
                              AMAT,-
                              UNIT,-
                              SPETBI,-
                              SPEMI,-
                              PMEAS,-
                    HGPS,-
                    HSTAR,-
                              AMAT,-
                              MINV3,-
                              MATAB,-
                              SPETBI,-
                              PMEAS,-
          UPDATE,-
          UNIT,-
          AMAT,-
          CMAT,-
          CRAISE,-
PLTDATA,-
MEAS,-
```

```
!        UTILITY FUNCTIONS
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
         VCRS,VDOT,VMAG,FQ
```

```
0001              SUBROUTINE CRAISE(N,COV,COVMIN,IPT)
            C*****************************************************************
            C        RAISES DIAGONAL ELEMENTS OF THE COVARIANCE MATRIX IF THEY DROP
            C        BELOW CERTAIN VALUES AND THEY ARE BEING ESTIMATED.
            C
            C        FORMAL PARAMETERS
            C            N = NUMBER OF ROWS OR COLS IN COVARIANCE MATRIX
            C            COV = COVARIANCE MATRIX
            C            COVMIN = MINIMUM VALUES FOR EACH VARIANCE
            C            IPT = ARRAY OF 1'S AND 0'S, A 1 INDICATING THAT A PARTICULAR
            C                  PARAMETER IS BEING ESTIMATED
            C*****************************************************************
            C        DIMENSION COV(N,N),COVMIN(N),IPT(N)
0002              ' DIMENSION COV(22,22),COVMIN(22),IPT(22)
0003              REAL*8 COV,COVMIN
0004              DO 100 I = 1,N
0005                  IF (IPT(I).EQ.0) GO TO 100
0006                  IF (COV(I,I).LT.COVMIN(I)) COV(I,I) = COVMIN(I)
0007         100      CONTINUE
0008              RETURN
0009              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	103	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	64	PIC CON REL LCL NOSHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		CRAISE

VARIABLES

Address	Type	Name	Address	Type	Name
2-00000000	I*4	I	AP-00000004@	I*4	N

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	COV	3872	(22, 22)
AP-0000000C@	R*8	COVMIN	176	(22)
AP-00000010@	I*4	IPT	88	(22)

```
LABELS

    Address   Label

  0-0000005F  100

Total Space Allocated = 167 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:          0.59 seconds
  Elapsed Time:      14.93 seconds
  Page Faults:       274
  Dynamic Memory:    160 pages
```

```
              SUBROUTINE CRESIF(XTRANS,XIN,XTRANP,XOUT)
      C
      C       THIS ROUTINE TRANSFORMS ERROR VECTORS FROM INERTIAL TO FLIGHT
      C
              REAL*8 XIN(3),XOUT(3),XTRANS(3,3),XTRANP(3,3),XTEMP(3)
      C
              DO 10 I=1,3
                RSUM=0
                DO 5 J=1,3
                  RSUM=RSUM+XTRANS(J,I)*XIN(J)
      5           CONTINUE
                XTEMP(I)=RSUM
      10        CONTINUE
      C
              DO 20 I=1,3
                RSUM=0
                DO 15 J=1,3
                  RSUM=RSUM+XTRANP(J,I)*XTEMP(J)
      15          CONTINUE
                XOUT(I)=RSUM
      20        CONTINUE
      C
              RETURN
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	169	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	116	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD

ENTRY POINTS

Address	Type	Name
0-00000000		CRESIF

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000018	I*4	I	2-00000020	I*4	J	2-0000001C	R*4	RSUM

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	XIN	24	(3)
AP-00000010@	R*8	XOUT	24	(3)
2-00000000	R*8	XTEMP	24	(3)

```
AP-0000000C@ R*8  XTRANP             72  (3, 3)
AP-00000004@ R*8  XTRANS             72  (3, 3)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
**	5	**	10	**	15	**	20

Total Space Allocated = 285 Bytes

```
      C
```

COMMAND QUALIFIERS

```
  FORTRAN CRESIF/LIS

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:            0.79 seconds
  Elapsed Time:        2.00 seconds
  Page Faults:         106
  Dynamic Memory:      28 pages
```

```
0001            SUBROUTINE CRTRAN(XTRANS,XIN,XOUT)
        C
        C       THIS ROUTINE TRANSFORMS ERROR VECTORS FROM INERTIAL TO FLIGHT
        C
0002            REAL*8 XIN(3),XOUT(3),XTRANS(3,3),RSUM
        C
0003            DO 10 I=1,3
0004              RSUM=0
0005              DO 5 J=1,3
0006                RSUM=RSUM+XTRANS(J,I)*XIN(J)
0007    5           CONTINUE
0008              XOUT(I)=RSUM
0009    10        CONTINUE
        C
0010            RETURN
0011            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	100	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	76	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD

ENTRY POINTS

Address	Type	Name
0-00000000		CRTRAN

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000008	I*4	I	2-0000000C	I*4	J	2-00000000	R*8	RSUM

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000008@	R*8	XIN	24	(3)
AP-0000000C@	R*8	XOUT	24	(3)
AP-00000004@	R*8	XTRANS	72	(3, 3)

LABELS

Address	Label	Address	Label
**	5	**	10

Total Space Allocated = 176 Bytes

```
      C

COMMAND QUALIFIERS

  FORTRAN CRTRAN/LIS

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:           0.54 seconds
  Elapsed Time:       2.24 seconds
  Page Faults:        99
  Dynamic Memory:     28 pages
```

```
        REAL*4 X(1000,6),TIME(1000),XARAY(1000)
        OPEN(UNIT=1,NAME='GAIN.DAT',TYPE='OLD',RECORDSIZE=98)
        I=1
2       CONTINUE
        READ(1,3,END=4)TIME(I),(X(I,J),J=1,6)
3       FORMAT(7F14.7)
        I=I+1
        GO TO 2
4       CONTINUE
        ITNUM=I-1
6       CONTINUE
        WRITE(5,10)
10      FORMAT(' INPUT PLOT NUMBER (-1=EXIT) :  ',$)
        READ(5,20)ITYPE
20      FORMAT(I5)
        IF(ITYPE.EQ.-1.OR.ITYPE.GT.6)GO TO 9000
        DO 30 I=1,ITNUM
          XARAY(I)=X(I,ITYPE)
30        CONTINUE
        CALL HP7221
        CALL QQHP(0.,0.,6)
        CALL LIMPLT(TIME,XARAY,ITNUM)
        GO TO 6
9000    CONTINUE
        END
```

```
PROGRAM SECTIONS

    Name                      Bytes   Attributes

 0 $CODE                        244   PIC CON REL LCL   SHR   EXE   RD NOWRT LONG
 1 $PDATA                        62   PIC CON REL LCL   SHR NOEXE   RD NOWRT LONG
 2 $LOCAL                     32080   PIC CON REL LCL NOSHR NOEXE   RD   WRT LONG

ENTRY POINTS

    Address   Type Name

 0-00000000      GAIN$MAIN

VARIABLES

    Address   Type Name          Address   Type Name          Address   Type Name          Address   Type Name

 2-00007D00   I*4 I          2-00007D08   I*4 ITNUM      2-00007D0C   I*4 ITYPE      2-00007D04   I*4 J

ARRAYS

    Address   Type Name                  Bytes  Dimensions

 2-00005DC0   R*4 TIME                    4000  (1000)
 2-00000000   R*4 X                      24000  (1000,6)
 2-00006D60   R*4 XARAY                   4000  (1000)

LABELS

    Address    Label     Address    Label     Address    Label     Address    Label     Address    Label     Address    Label

 0-00000014   2       1-00000011   3'     0-00000067   4      0-0000006C   6      1-00000017   10'    1-0000003B   20'
      **      30      0-000000F0   9000

FUNCTIONS AND SUBROUTINES REFERENCED

FOR$OPEN        HP7221        LINPLT        QQHP

Total Space Allocated = 32386 Bytes

COMPILER OPTIONS

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/OPTIMIZE  /WARNINGS  /I4  /NOD_LINES
```

6-Apr-1981 14:50:49   VAX-11 FORTRAN V2.0-2   Page 1
25-Mar-1981 09:00.45   _DBA0:[D11R.GCH]GAUSS.FOR;11

```
0001            FUNCTION GAUSS(MEAN,SIGMA)
          C**********************************************************************
          C         RANDOM NOISE GENERATOR USING PARAMETER MEAN AND STANDARD DEV
          C**********************************************************************
0002            INCLUDE 'DEBUG.COM'
0003    *       COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                 IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                 IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
0004            REAL*8 MEAN,SIGMA,GAUSS,RAN
0005            DATA I1,I2/12345,54321/
0006            GAUSS = 0.
0007            DO 10 I=1,12
0008      10    GAUSS = GAUSS + RAN(I1,I2)
0009            GAUSS = MEAN + SIGMA*(GAUSS-6.)
0010            RETURN
0011            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	52	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	32	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000	R*8	GAUSS

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000010	I*4	I	2-00000008	I*4	I1	2-0000000C	I*4	I2	3-00000004	I*4	IDEBUG
3-00000000	I*4	IENTER	AP-00000004@	R*8	MEAN	AP-00000008@	R*8	SIGMA			

LABELS

Address	Label
**	10

FUNCTIONS AND SUBROUTINES REFERENCED

FOR$IRAN

Total Space Allocated = 92 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:            0.58 seconds
  Elapsed Time:        6.23 seconds
  Page Faults:         337
  Dynamic Memory:      160 pages
```

```
GCPPLOT/CO:80,-
INEFLT,-
TRANFM,-
CRTRAN,-
CRESIF,-
MINV3,-
UNIT,-
VCRS,-
VDOT,-
VMAG
```

```
          C
          C--- GCPPLOT (GCP PLOT ROUTINES)
          C
          C
0001                INCLUDE 'ARRAY.COM'
        * C         COMMON BLOCK USED FOR PLOTTING ROUTINE
        * C
0002    *           COMMON /ARRAY/ X(3000,39,2), INUM
0003    *           REAL*4 X,X1(3000,39),X2(3000,39)
0004    *           INTEGER*2 INUM
0005    *           EQUIVALENCE (X1(1,1),X(1,1,1)),(X2(1,1),X(1,1,2))
        * C
0006                CHARACTER*64 MENUS(23,6)
0007                CHARACTER*20 XTIT,YTIT(4)
0008                CHARACTER*40 XREF(2)
0009                CHARACTER*50 CTITLE(38)
0010                LOGICAL*1    ATYPE,AXTIT(20),AYTIT(80),ATITLE(1900),ABELLS(10)
0011                LOGICAL*1    ATIT(64),AOTYPE,AREF(80),AFRAME(2)
0012                INTEGER*2    IOPTS(4),ITYPE,IMAX(4),IOFF(4),IOPTOF(38),IFRAME
0013                REAL*4       TIME(3000),YARRAY(12000)
0014                EQUIVALENCE (AXTIT(1),XTIT),(AYTIT(1),YTIT(1))
0015                EQUIVALENCE (ATITLE(1),CTITLE(1)),(AREF(1),XREF(1))
0016                DATA IFRAME/1/
0017                DATA ABELLS/7,7,7,7,7,7,7,7,7,7/
0018                DATA IOPTOF/2,3,4,2,12,13,14,12,5,6,7,5,15,16,17,15,9,10,11,9,
               *            19,20,21,19,22,23,24,25,26,27,28,29,30,28,31,32,33,
               *            31/
0019                DATA IMAX/8,8,8,14/
0020                DATA IOFF/0,8,16,24/
          C
          C
0021                DATA MENUS/' ',' ',' ',' ',
               *            '  W E L C O M E   to the GCP simulation plot ',
               *            '                  routines. ',' ',' ',' ',
               *            ' These plot routines allow the user to overlay up ',
               *            ' to 4 plots as long as the plots to be generated ',
               *            ' are of the same units. The user will be prompted ',
               *            ' by the command OPTION>> or by the sentence Hit   ',
               *            ' RETURN to continue >> . The user has the option ',
               *            ' of using the HP7221 Hewlett Packard plotter (H)  ',
               *            ' or the 4014 Tectronix plotter (T). If the user is',
               *            ' using the HP7221, before this task may be run he ',
               *            ' must type in ASSIGN TTD7: FOR010 so that the     ',
               *            ' HP7221 will be properly assigned.                ',
               *            ' ',' ',' ',' ',
               *            ' ',' ',' ',' ',' ',
               *            ' The following are the type(s) of plot(s) which   ',
               *            ' may be generated :  ',' ',' ',
               *            '                  1    POSITION ERRORS (KM)       ',
               *            '                  2    VELOCITY ERRORS (KM/SEC)   ',
               *            '                  3    ATTITUDE ERRORS (ARC SEC)  ',
               *            '                  4    RESIDUALS (KM OR KM/SEC)   ',
               *            '                 -1    EXIT PLOT ROUTINES         ',
               *            ' ',' ',' ',' ',' ',' ',' ',' ',' ',
               *            ' ',' ',' ',' ',' ',
               *            ' POSITION ERROR PLOTS : ',' ',' ',
               *            '                  1    NAVIGATION X POSITION      ',
```

```
        *               '                2    NAVIGATION Y POSITION        ',
        *               '                3    NAVIGATION Z POSITION        ',
        *               '                4    RSS POSITION ERROR           ',
        *               '                5    STANDARD DEV. X POSITION     ',
        *               '                6    STANDARD DEV. Y POSITION     ',
        *               '                7    STANDARD DEV. Z POSITION     ',
        *               '                8    RSS STANDARD DEV. POSITION   ',
        *               ' ',' ',' ',' ',' ',' ',' ',' ',
        *               ' ',' ',' ',' ',' ',
        *               ' VELOCITY ERROR PLOTS : ',' ',' ',
        *               '                1    NAVIGATION X VELOCITY        ',
        *               '                2    NAVIGATION Y VELOCITY        ',
        *               '                3    NAVIGATION Z VELOCITY        ',
        *               '                4    RSS VELOCITY ERROR           ',
        *               '                5    STANDARD DEV. X VELOCITY     ',
        *               '                6    STANDARD DEV. Y VELOCITY     ',
        *               '                7    STANDARD DEV. Z VELOCITY     ',
        *               '                8    RSS STANDARD DEV. VELOCITY   ',
        *               ' ',' ',' ',' ',' ',' ',' ',' ',
        *               ' ',' ',' ',' ',' ',
        *               ' ATTITUDE ERROR PLOTS : ',' ',' ',
        *               '                1    ROLL                         ',
        *               '                2    PITCH                        ',
        *               '                3    YAW                          ',
        *               '                4    RSS (ROLL, PITCH AND YAW)    ',
        *               '                5    STANDARD DEV. ROLL           ',
        *               '                6    STANDARD DEV. PITCH          ',
        *               '                7    STANDARD DEV. YAW            ',
        *               '                8    STANDARD DEV. RSS            ',
        *               ' ',' ',' ',' ',' ',' ',' ',' ',
        *               ' ',' ',' ',
        *               ' RESIDUALS : ',' ',' ',
        *               '            1    L.M. TRACKER HORIZONTAL  (KM)     ',
        *               '            2    L.M. TRACKER VERTICAL    (KM)     ',
        *               '            3    STAR TRACKER 1 HORZ.     (KM)     ',
        *               '            4    STAR TRACKER 1 VERT.     (KM)     ',
        *               '            5    STAR TRACKER 2 HORZ.     (KM)     ',
        *               '            6    STAR TRACKER 2 VERT.     (KM)     ',
        *               '            7    GPS X POSITION           (KM)     ',
        *               '            8    GPS Y POSITION           (KM)     ',
        *               '            9    GPS Z POSITION           (KM)     ',
        *               '           10    GPS RSS POSITION         (KM)     ',
        *               '           11    GPS X VELOCITY           (KM/SEC) ',
        *               '           12    GPS Y VELOCITY           (KM/SEC) ',
        *               '           13    GPS Z VELOCITY           (KM/SEC) ',
        *               '           14    GPS RSS VELOCITY         (KM/SEC) ',
        *               ' ',' ',' '/
0022          DATA XTIT/' TIME (SECONDS)'/
0023          DATA YTIT/' UNITS  (KM) ',
        *               ' UNITS  (KM/SEC.)',
        *               ' UNITS  (ARC SEC.)',
        *               ' UNITS  (RADIANS) '/
0024          DATA XREF/' *** INERTIAL COORDINATE FRAME ***',
        *               ' *** FLIGHT COORDINATE FRAME *** '/
0025          DATA CTITLE /' NAVIGATION X POSITION ERROR
        *                 ' NAVIGATION Y POSITION ERROR
        *                 ' NAVIGATION Z POSITION ERROR
```

```
     *                ' NAVIGATION POSITION ERROR RSS                      ',
     *                ' STANDARD DEV. X POSITION ERROR                     ',
     *                ' STANDARD DEV. Y POSITION ERROR                     ',
     *                ' STANDARD DEV. Z POSITION ERROR                     ',
     *                ' STANDARD DEV. POSITION ERROR RSS                   ',
     *                ' NAVIGATION X VELOCITY ERROR                        ',
     *                ' NAVIGATION Y VELOCITY ERROR                        ',
     *                ' NAVIGATION Z VELOCITY ERROR                        ',
     *                ' NAVIGATION VELOCITY ERROR RSS                      ',
     *                ' STANDARD DEV. X VELOCITY ERROR                     ',
     *                ' STANDARD DEV. Y VELOCITY ERROR                     ',
     *                ' STANDARD DEV. Z VELOCITY ERROR                     ',
     *                ' STANDARD DEV. VELOCITY ERROR RSS                   ',
     *                ' ATTITUD  ERROR ROLL                                ',
     *                ' ATTITUD  ERROR PITCH                               ',
     *                ' ATTITUD  ERROR YAW                                 ',
     *                ' ATTITUD  ERROR RSS                                 ',
     *                ' STANDARD DEV. ATTITUDE ERROR ROLL                  ',
     *                ' STANDARD DEV. ATTITUDE ERROR PITCH                 ',
     *                ' STANDARD DEV. ATTITUDE ERROR YAW                   ',
     *                ' STANDARD DEV. ATTITUDE ERROR RSS                   ',
     *                ' L.M. TRACKER RESIDUAL HORIZONTAL                   ',
     *                ' L.M. TRACKER RESIDUAL VERTICAL                     ',
     *                ' STAR TRACKER 1 RESIDUAL HORIZONTAL                 ',
     *                ' STAR TRACKER 1 RESIDUAL VERTICAL                   ',
     *                ' STAR TRACKER 2 RESIDUAL HORIZONTAL                 ',
     *                ' STAR TRACKER 2 RESIDUAL VERTICAL                   ',
     *                ' GPS X POSITION RESIDUAL                            ',
     *                ' GPS Y POSITION RESIDUAL                            ',
     *                ' GPS Z POSITION RESIDUAL                            ',
     *                ' GPS RSS POSITION RESIDUAL                          ',
     *                ' GPS X VELOCITY RESIDUAL                            ',
     *                ' GPS Y VELOCITY RESIDUAL                            ',
     *                ' GPS Z VELOCITY RESIDUAL                            ',
     *                ' GPS RSS VELOCITY RESIDUAL                          '/
C
C
C*********************************************************************
C                                                                    *
C--- DEFINITION OF X                                                 *
C
C      X(I,1)         =  TIME
C      X(I,2-4)       =  X,Y,Z POSITION ERRORS
C      X(I,5-7)       =  X,Y,Z VELOCITY ERRORS
C      X(I,8-11)      =  QUATERNION ATTITUDE ERRORS
C      X(I,12-14)     =  STANDARD DEV. X,Y,Z POSITION ERRORS
C      X(I,15-17)     =  STANDARD DEV. X,Y,Z VELOCITY ERRORS
C      X(I,18-21)     =  STANDARD DEV. QUATERNION ATTITUDE ERRORS
C      X(I,22-23)     =  L.M. HORZ.,VERT. RESIDUALS
C      X(I,24-25)     =  STAR TRACKER 1 HORZ.,VERT. RESIDUALS
C      X(I,26-27)     =  STAR TRACKER 2 HORZ.,VERT. RESIDUALS
C      X(I,28-30)     =  GPS X,Y,Z POSITION RESIDUALS
C      X(I,31-33)     =  GPS X,Y,Z VELOCITY RESIDUALS
C      X(n,n,1)       =  COORDINATES IN INERTIAL FRAME
C      X(n,n,2)       =  COORDINATES IN FLIGHT FRAME
C
C*********************************************************************
```

```
C
C
C
C--- START PLOTS
C
C
C--- GET OUTPUT PLOTTING DEVICE TYPE
C
        WRITE(5,908)
        READ(5,903)AOTYPE
C
C--- OPEN INPUT DATA FILE AND READ THE PLOT TITLE
C
        OPEN(UNIT=2,NAME='PLOT.DAT',TYPE='OLD',RECORDSIZE=780)
        READ(2,907)(ATIT(I),I=1,64)
        I=1
 10     CONTINUE
        READ(2,900,END=20)(X(I,J,1),J=1,39)
        I=I+1
        GO TO 10
 20     CONTINUE
        INUM=I-1
        WRITE(5,2)INUM
 2      FORMAT(' TOTAL POINTS READ IN : ',I4)
C
C--- WRITE OUT FIRST MENU
C
        WRITE(5,901)(MENUS(I,1),I=1,23)
        WRITE(5,912)
        WRITE(5,913)
        READ(5,903)AFRAME(1)
        WRITE(5,914)
        READ(5,903)AFRAME(2)
C
C--- SET UP FLIGHT COORDINATE SIDE OF X IF NECESSARY
C
        IF(AFRAME(2).NE.'Y')GO TO 25
          WRITE(5,915)
          CALL INEFLT
 25       CONTINUE
C
C--- PUT UP SECOND MENU
C
 30     CONTINUE
        WRITE(5,901)(MENUS(I,2),I=1,23)
        WRITE(5,904)
        READ(5,905)ITYPE
C
C--- ERROR CHECK ON ITYPE
C
        IF(ITYPE.EQ.-1)GO TO 9999
C
          IF(ITYPE.GT.0.AND.ITYPE.LE.4)GO TO 35
            WRITE(5,920)ABELLS
            WRITE(5,902)
            READ(5,903)ATYPE
            GO TO 30
```

```
0059    35          CONTINUE
        C
        C--- PUT UP CORRECT MENU, AND RECIEVE OPTION(S)
        C
0060    40      CONTINUE
                WRITE(5,901)(MENUS(I,ITYPE+2),I=1,23)
0062            WRITE(5,906)
0063            READ(5,905)(IOPTS(I),I=1,4)
0064    43      CONTINUE
0065            IF(ITYPE.EQ.3.OR.AFRAME(2).NE.'Y')GO TO 45
0066              WRITE(5,916)
0067              READ(5,917)IFRAME
0068              IF(IFRAME.LT.1.OR.IFRAME.GT.2)GO TO 43
0069    45        CONTINUE
        C
        C--- ERROR CHECK IOPT
        C
0070            DO 60 I=1,4
0071              IF(IOPTS(I).GE.0.AND.IOPTS(I).LE.IMAX(ITYPE))GO TO 50
0072                WRITE(5,920)ABELLS
0073                WRITE(5,902)
0074                READ(5,903)
0075                GO TO 40
0076    50          CONTINUE
0077    60        CONTINUE
        C
        C--- DETERMINE THE NUMBER OF PLOT OVERLAYS
        C
0078            ITOT=0
0079            DO 70 I=1,4
0080              IF(IOPTS(I).NE.0)ITOT=ITOT+1
0081    70        CONTINUE
        C
        C--- DETERMINE THE PLOTS UNITS
        C
0082            IUNIT=1
0083            IF(ITYPE.EQ.2)IUNIT=2
0084            IF(ITYPE.EQ.3)IUNIT=3
        C
        C--- IF THIS IS A RESIDUAL PLOT, CHECK FOR UNITS CONFORMITY
        C
        C--- CHECK FOR EMBEDDED ZERO IN OPTION STRING
        C
0085            IF(ITOT.EQ.4)GO TO 68
0086              DO 65 I=ITOT+1,4
0087                IF(IOPTS(I).EQ.0)GO TO 63
0088                  WRITE(5,922)ABELLS
0089                  WRITE(5,902)
0090                  READ(5,903)ATYPE
0091                  GO TO 40
0092    63            CONTINUE
0093    65          CONTINUE
0094    68        CONTINUE
        C
0095            IERR=0
0096            IF(ITYPE.NE.4)GO TO 80
0097              IT=1
```

```
                  IF(IOPTS(1).GT.10)IT=2
                  IF(IT.EQ.2)GO TO 75
                    DO 73 I=1,ITOT
                      IF(IOPTS(I).GT.10)IERR=1
         73           CONTINUE
                    IUNIT=1
                    GO TO 78
         75         CONTINUE
                    DO 77 I=1,ITOT
                      IF(IOPTS(I).LT.11)IERR=1
         77           CONTINUE
                    IUNIT=2
         78         CONTINUE
         80       CONTINUE
                IF(ITYPE.EQ.4.AND.(IOPTS(1).EQ.1.OR.IOPTS(1).EQ.2))IUNIT=4
                WRITE(5,1)IUNIT
          1     FORMAT(' IUNIT= ',I10)
                IF(IERR.EQ.0)GO TO 90
                  WRITE(5,921)ABELLS
                  WRITE(5,902)
                  READ(5,903)ATYPE
                  GO TO 40
         90       CONTINUE
        C
        C--- EVERYTHING CHECKS OUT GOOD SO FAR
        C
        C
        C--- GET THE TIME FRAME USED FOR THIS PLOT
        C
        100     CONTINUE
                WRITE(5,909)X(1,1,1),X(INUM,1,1)
                READ(5,910)RSTART,RSTOP
                IF(RSTART.LT.RSTOP)GO TO 110
                  WRITE(5,911)
                  GO TO 100
        110       CONTINUE
                IF(RSTOP.GT.X(INUM,1,1))RSTOP=X(INUM,1,1)
                IF(RSTART.LT.X(1,1,1)) RSTART=X(1,1,1)
        C
        C--- GET THE BOUNDS ON THE TIMES
        C
                DO 120 I=1,INUM-1
                  IF(RSTART.GE.X(I,1,1).AND.RSTART.LE.X(I+1,1,1))GO TO 130
        120       CONTINUE
        130       CONTINUE
                ITSTRT=I
                DO 140 I=1,INUM-1
                  IF(RSTOP.GE.X(I,1,1).AND.RSTOP.LE.X(I+1,1,1))GO TO 150
        140       CONTINUE
        150       CONTINUE
                ITSTOP=I+1
                ITNUM=ITSTOP-ITSTRT+1
        C
        C--- SET UP TIME ARRAY
        C
                DO 160 I=1,ITNUM
                  TIME(I)=X(I+ITSTRT-1,1,1)
```

```
 160        CONTINUE
C*************************************************************
C    CREATE THE PLOTS                                         *
C*************************************************************
C
C--- INITIALIZE THE PLOTTING DEVICE
C
            IF(AOTYPE.EQ.'T')GO TO 170
              CALL HP7221
              CALL QQHP(0.,0.,6)                     ! FIRST ARG IS PEN TYPE
C                                                      0. = BLACK
C                                                      1. = RED
C                                                      2. = BLUE
C                                                      3. = GREEN
              GO TO 180
 170          CONTINUE
              CALL TK
 180          CONTINUE
            CALL PAGE(8.5,11.)
            CALL PHYSOR(1.5,1.)
            CALL TITLE(' ',1,AXTIT,20,AYTIT((IUNIT-1)*20+1),20,6.,8.)
C
C--- PUT UP TITLE AND LEGEND
C
            CALL HEIGHT(.20)
            CALL MESSAG(ATIT,40,0.,9.7)
            WRITE(5,828)IFRAME
 828        FORMAT(' IFRAME = ',I6)
            CALL MESSAG(AREF((IFRAME-1)*40+1),40,0.,9.4)
            CALL HEIGHT(.15)
            DO 190 I=1,ITOT
              ITEMP=IOFF(ITYPE)+IOPTS(I)
              IF(AOTYPE.EQ.'H')CALL QQHP(FLOAT(I-1),0.,6)
              CALL MESSAG(ATITLE((ITEMP-1)*50+1),50,2.,9.-(I-1)*.25)
              GO TO (182,184,186,188)I
 182          CONTINUE
              CALL RESET('DOT')
              GO TO 189
 184          CONTINUE
              CALL DASH
              GO TO 189
 186          CONTINUE
              CALL DOT
              GO TO 189
 188          CONTINUE
              CALL CHNDOT
 189          CONTINUE
              CALL VECTOR(0.,9.-(I-1)*.25+.08,1.5,9.-(I-1)*.25+.08,0)
 190          CONTINUE
            CALL RESET('CHNDOT')
            CALL QQHP(0.,0.,6)
C
C--- PROCESS CURVES NOW. STOR. VALUES INTO YARRAY
C
            DO 600 I=1,ITOT
              IPTR=IOFF(ITYPE)+IOPTS(I)
              ITEMP=IOPTOF(IPTR)
```

```
        C
        C--- GO TO THE PROPER OPTION
        C
0184          GO TO (200,200,200,240,200,200,200,240,200,200,200,240,
             *       200,200,200,240,300,300,300,340,300,300,300,340,
             *       200,200,200,200,200,200,200,200,200,240,200,200,
             *       200,240)IPTR
0185    200   CONTINUE
        C
        C--- PLOTS REQUIRING NO TRANSLATION OF DATA
        C
0186          DO 210 J=1,ITNUM
0187            YARRAY((I-1)*3000+J)=X(J+ITSTRT-1,ITEMP,IFRAME)
0188    210     CONTINUE
0189          GO TO 500
0190    240   CONTINUE
        C
        C--- RSS PLOTS REQUIRING NO TRANSLATION OF DATA
        C
0191          DO 250 J=1,ITNUM
0192            YARRAY((I-1)*3000+J)=SQRT(X(J+ITSTRT-1,ITEMP,IFRAME)**2+
             *                            X(J+ITSTRT-1,ITEMP+1,IFRAME)**2+
             *                            X(J+ITSTRT-1,ITEMP+2,IFRAME)**2)
0193    250     CONTINUE
0194          GO TO 500
0195    300   CONTINUE
        C
        C--- ATTITUDE ERRORS. CONVERT TO ARC SECONDS.
        C
0196          DO 310 J=1,ITNUM
0197            YARRAY((I-1)*3000+J)=(X(J+ITSTRT-1,ITEMP,IFRAME)*2.)/.4848E-5
0198    310     CONTINUE
0199          GO TO 500
0200    340   CONTINUE
        C
        C--- ATTITUDE RSS
        C
0201          DO 350 J=1,ITNUM
0202            YARRAY((I-1)*3000+J)=(2.*SQRT(X(J+ITSTRT-1,ITEMP,IFRAME)**2+
             *                       X(J+ITSTRT-1,ITEMP+1,IFRAME)**2+
             *                       X(J+ITSTRT-1,ITEMP+2,IFRAME)**2))
             *                                 /.4848E-5
0203    350     CONTINUE
0204    500   CONTINUE
0205    600   CONTINUE
        C
        C--- FIND MIN AND MAX.
        C
0206        RMAX=-300000000.
0207        RMIN=300000000.
0208        DO 700 I=1,ITOT
0209          DO 650 J=1,ITNUM
0210            IF(YARRAY((I-1)*3000+J).LT.RMIN)RMIN=YARRAY((I-1)*3000+J)
0211            IF(YARRAY((I-1)*3000+J).GT.RMAX)RMAX=YARRAY((I-1)*3000+J)
0212    650     CONTINUE
0213    700   CONTINUE
        C
```

```
C--- PLOT GRID AND AXIS
C
        CALL GRAF(TIME(1),'SCALE',TIME(ITNUM),RMIN,'SCALE',RMAX)
        CALL GRID(2,2)
C
C--- DRAW CURVES
C
        DO 800 I=1,ITOT
          CALL QQHP(FLOAT(I-1),0.,6)
          GO TO (710,720,730,740)I
710       CONTINUE
          CALL RESET('DOT')
          GO TO 750
720       CONTINUE
          CALL DASH
          GO TO 750
730       CONTINUE
          CALL DOT
          GO TO 750
740       CONTINUE
          CALL CHNDOT
750       CONTINUE
          CALL CURVE(TIME,YARRAY((I-1)*3000+1),ITNUM,0)
800       CONTINUE
        CALL ENDPL(0)
        GO TO 30
9999    CONTINUE
900     FORMAT(39F20.10)
901     FORMAT(A64)
902     FORMAT(1X,' Hit RETURN to continue ',$)
903     FORMAT(A1)
904     FORMAT(' OPTION >> ',$)
905     FORMAT(4I4)
906     FORMAT(' OPTION(S) >> ',$)
907     FORMAT(64A1)
908     FORMAT  (' PLOTTER TYPE (T OR H) : ',$)
909     FORMAT(' INPUT START,STOP TIMES FOR THE PLOTS. ',/,
     *         ' MIN, MAX TIMES ARE : ',2(F6.1,2X),' TIMES : ',$)
910     FORMAT(2F6.1)
911     FORMAT(' BAD START,STOP TIMES. START MUST BE LESS THEN STOP')
912     FORMAT(' INPUT REFERENCE FRAMES PLOTS WILL BE GENERATED IN :')
913     FORMAT('        INERTIAL COORDINATES (Y OR N) ?  ',$)
914     FORMAT('        FLIGHT   COORDINATES (Y OR N) ?  ',$)
915     FORMAT('             ... one moment please ')
916     FORMAT(' REFERENCE FRAME TYPE FOR PLOT  ',/,
     *         '   INERTIAL    =    1 ',/,
     *         '   FLIGHT      =    2 ',/,
     *         ' REFERNCE TYPE : ',$)
917     FORMAT(I1)
920     FORMAT(////////////,' INVALID OPTION TYPE ',10A1,//////////)
921     FORMAT(////////////,' ILLEGAL MIXING OF UNITS',10A1,//////////)
922     FORMAT(////////////,' ILLEGAL EMBEDDED ZERO IN OPTION STRING',
     *         10A1,//////////)
        END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	2990	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	878	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	71544	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 ARRAY	936002	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		GCPPLOT$MAIN

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-000115B1	L*1	AOTYPE	2-000115B0	L*1	ATYPE	2-000115B8	I*4	I	2-000115C8	I*4	IERR
2-000115B4	I*2	IFRAME	3-000E4840	I*2	INUM	2-000115E8	I*4	IPTR	2-000115CC	I*4	IT
2-000115E4	I*4	ITEMP	2-000115E0	I*4	ITNUM	2-000115C0	I*4	ITOT	2-000115DC	I*4	ITSTOP
2-000115D8	I*4	ITSTRT	2-000115B2	I*2	ITYPE	2-000115C4	I*4	IUNIT	2-000115BC	I*4	J
2-000115EC	R*4	RMAX	2-000115F0	R*4	RMIN	2-000115D0	R*4	RSTART	2-000115D4	R*4	RSTOP
2-0000080C	CHAR	XTIT									

A-327

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00011564	L*1	ABELLS	10	(10)
2-000115AE	L*1	AFRAME	2	(2)
2-00000000	L*1	AREF	80	(80)
2-0001156E	L*1	ATIT	64	(64)
2-00000050	L*1	ATITLE	1900	(1900)
2-0000080C	L*1	AXTIT	20	(20)
2-000007BC	L*1	AYTIT	80	(80)
2-00000050	CHAR	CTITLE	1900	(38)
2-0000F288	I*2	IMAX	8	(4)
2-0000F290	I*2	IOFF	8	(4)
2-0000F298	I*2	IOPTOF	76	(38)
2-0000F280	I*2	IOPTS	8	(4)
2-0000F2E4	CHAR	MENUS	8832	(23, 6)
2-00000820	R*4	TIME	12000	(3000)
3-00000000	R*4	X	936000	(3000, 39, 2)
3-00000000	R*4	X1	468000	(3000, 39)
3-00072420	R*4	X2	468000	(3000, 39)
2-00000000	CHAR	XREF	80	(2)
2-00003700	R*4	YARRAY	48000	(12000)
2-000007BC	CHAR	YTIT	80	(4)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-0000001D	1'	1-00000000	2'	0-00000076	10	0-00000088	20	0-000001C9	25	0-000001C9	30
0-000002A9	35	0-000002A9	40	0-0000032E	43	0-00000380	45	0-000003CA	50	**	60
0-0000048E	63	**	65	0-00000492	68	**	70	**	73	0-000004CF	75
**	77	0-000004F0	78	0-000004F4	80	0-00000589	90	0-00000590	100	0-0000060A	110
**	120	0-0000064F	130	**	140	0-00000685	150	**	160	0-000006D6	170
0-000006DD	180	0-000007EB	182	0-000007F6	184	0-000007FF	186	0-00000808	188	0-0000080F	189
**	190	0-000008DF	200	**	210	0-0000092E	240	**	250	0-000009AF	300
**	310	0-00000A07	340	**	350	0-00000A8D	500	**	600	**	650
**	700	0-00000856	710	0-00000861	720	0-0000086A	730	0-00000873	740	0-0000087A	750
**	800	1-0000002A	828'	1-00000039	900'	1-0000003F	901'	1-00000042	902'	1-00000060	903'
1-00000063	904'	1-00000072	905'	1-00000077	906'	1-00000089	907'	1-0000008E	908'	1-000000AB	909'
1-00000104	910'	1-0000010A	911'	1-00000140	912'	1-00000177	913'	1-000001A3	914'	1-000001CF	915'
1-000001F2	916'	1-00000258	917'	1-00000258	920'	1-0000028D	921'	1-000002C2	922'	0-000008AA	9999

FUNCTIONS AND SUBROUTINES REFERENCED

CHNDOT	CURVE	DASH	DOT	ENDPL	FOR$OPEN	GRAF	GRID
HEIGHT	HP7221	INEFLT	MESSAG	MTH$SQRT	PAGE	PHYSOR	QQHP
RESET	TITLE	TK	VECTOR				

Total Space Allocated = 1011414 Bytes

COMMAND QUALIFIERS

```
FORTRAN /LIS/CO:77 GCPPLOT

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=77
```

COMPILATION STATISTICS

```
Run Time:            9.48 seconds
Elapsed Time:        54.60 seconds
Page Faults:         592
Dynamic Memory:      143 pages
```

```
            SUBROUTINE INEFLT
C
C--- THIS ROUTINE TRANFORMS THINGS IN X FROM INERTIAL TO FLIGHT COORDINATES.
C
            INCLUDE 'ARRAY.COM'
C           COMMON BLOCK USED FOR PLOTTING ROUTINE
C
            COMMON /ARRAY/ X(3000,39,2), INUM
            REAL*4 X,X1(3000,39),X2(3000,39)
            INTEGER*2 INUM
            EQUIVALENCE (X1(1,1),X(1,1,1)),(X2(1,1),X(1,1,2))
C
            REAL*8 XTRANS(3,3,3000),XTRANP(3,3,3000)
            REAL*8 XTEMP1(3),XTEMP2(3)
C
C
C--- CREATE TRANFORMATION ARRAYS
C
            CALL TRANFM(XTRANS,XTRANP)
C
C--- TRANSFER THINGS IN X THAT WILL NOT BE TRANFORMED
C
            DO 10 I=1,INUM
               X(I,1,2)=X(I,1,1)
               X(I,18,2)=X(I,18,1)
               X(I,19,2)=X(I,19,1)
               X(I,20,2)=X(I,20,1)
               X(I,21,2)=X(I,21,1)
               X(I,22,2)=X(I,22,1)
               X(I,23,2)=X(I,23,1)
               X(I,24,2)=X(I,24,1)
               X(I,25,2)=X(I,25,1)
               X(I,26,2)=X(I,26,1)
               X(I,27,2)=X(I,27,1)
 10            CONTINUE
C
C--- CREATE TRANSFORMED  ERRORS
C
            DO 20 I=1,INUM
               XTEMP1(1)=X(I,2,1)
               XTEMP1(2)=X(I,3,1)
               XTEMP1(3)=X(I,4,1)
               CALL CRTRAN(XTRANS,XTEMP1(1),XTEMP2(1))
               X(I,2,2)=XTEMP2(1)
               X(I,3,2)=XTEMP2(2)
               X(I,4,2)=XTEMP2(3)
               XTEMP1(1)=X(I,5,1)
               XTEMP1(2)=X(I,6,1)
               XTEMP1(3)=X(I,7,1)
               CALL CRTRAN(XTRANS,XTEMP1(1),XTEMP2(1))
               X(I,5,2)=XTEMP2(1)
               X(I,6,2)=XTEMP2(2)
               X(I,7,2)=XTEMP2(3)
 20            CONTINUE
C
C--- CREATE RESIDUALS TRANSFORMED
C
```

```
                DO 30 I=1,INUM
      C           CALL CRESIF(XTRANS(1,1,I),X(I,12,1),XTRANP(1,1,I),X(I,12,2))
      C           CALL CRESIF(XTRANS(1,1,I),X(I,15,1),XTRANP(1,1,I),X(I,15,2))
      C           CALL CRESIF(XTRANS(1,1,I),X(I,28,1),XTRANP(1,1,I),X(I,28,2))
      C           CALL CRESIF(XTRANS(1,1,I),X(I,31,1),XTRANP(1,1,I),X(I,31,2))
      30          CONTINUE
                RETURN
                END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	350	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	432080	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3 ARRAY	936002	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		INEFLT

VARIABLES

Address	Type	Name	Address	Type	Name
2-00069780	I*4	I	3-000E4840	I*2	INUM

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-00000000	R*4	X	936000	(3000, 39, 2)
3-00000000	R*4	X1	468000	(3000, 39)
3-00072420	R*4	X2	468000	(3000, 39)
2-00069780	R*8	XTEMP1	24	(3)
2-00069798	R*8	XTEMP2	24	(3)
2-00034BC0	R*8	XTRANP	216000	(3, 3, 3000)
2-00000000	R*8	XTRANS	216000	(3, 3, 3000)

LABELS

Address	Label	Address	Label	Address	Label
**	10	**	20	**	30

```
FUNCTIONS AND SUBROUTINES REFERENCED

CRTRAN         TRANFM


Total Space Allocated = 1368432 Bytes


COMMAND QUALIFIERS

  FORTRAN /LIS INEFLT

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19


COMPILATION STATISTICS

  Run Time:            1.46 seconds
  Elapsed Time:        3.50 seconds
  Page Faults:         168
  Dynamic Memory:      51 pages
```

```
$!LINKUP.COM
$!
$!      LINKUP.COM LINKS ALL THE NECESSARY ROUTINES FOR EXECUTION
$!      OF THE GCP SIMULATION PROGRAM
$!
$LINK/MAP=GCP.MAP/FULL/CROSS_REFERENC_ -     !/DEB-
        GCP,-
                INDATA,-
                        MATAB,-
                OUTDATA,-
                        RUNG,-
                                DNAV,-
                                        EPHEM,-
                                                TRUEA,-
                                        SPRESS,-
                                                OCCULT,-
                                        GPERT,-
                        MATAB,-
                DNAV,-
                        EPHEM,-
                                TRUEA,-
                        SPRESS,-
                                OCCULT,-
                        GPERT,-
                GCPSEQ,-
                        VISIBLE,-
                                OCCULT,-
                GENENV,-
                        TREG,-
                        GYROUT,-
                                RATE,-
                                BMAT,-
                                CMAT,-
                                MATAB,-
                                MINV3,-
                                GAUSS,-
                                KATT,-
                                        BMAT,-
                                        CMAT,-
                                        MATAB,-
                                        QMULT,-
                                        UNIT,-
                        RUNG,-
                                DNAV,-
                                        EPHEM,-
                                                TRUEA,-
                                        SPRESS,-
                                                OCCULT,-
                                        GPERT,-
                GYRO,-
                        KATT,-
                                BMAT,-
                                CMAT,-
                                MATAB,-
                                QMULT,-
                                UNIT,-
                        PDATT,-
                                MATAB,-
                        PATT,-
                MEASURE,-
                        RANDU,-
                        CLCTBL -
```

```
                        RDGCP.-
                                RDDTA.-
!                               AMAT.-
!                               MATAB.-
!                               MINV3.-
!                               WET.-
                        BVECT.-
!                               MATAB.-
                                AMAT.-
!                       VISIBLE.-
!                               OCCULT.-
                        LAMKT.-
                                WET.-
!                                       MATAB.-
!                               MATAB.-
!                               AMAT.-
!                               MINV3.-
!                               UNIT.-
!                               GAUSS.-
                        GPS.-
!                               GAUSS.-
                        START.-
!                               GAUSS.-
                INTG.-
!                       TREG.-
                        RKG.-
!                               DNAV.-
!                                       EPHEM.-
!                                               TRUEA.-
!                                       SPRESS.-
!                                               OCCULT.-
!                                       GPERT.-
                                PDNAV.-
!                                       MATAB.-
                EST.-
                        PROP.-
                        MPART.-
                                HLMT.-
!                                       WET.-
!                                               MATAB.-
!                                       MATAB.-
!                                       MINV3.-
!                                       AMAT.-
!                                       UNIT.-
                                        SPETBI.-
                                        SPEV.-
                                        PMEAS.-
                                HGPS.-
                                HSTAR.-
!                                       AMAT.-
!                                       MINV3.-
!                                       MATAB.-
!                                       SPETBI.-
!                                       PMEAS.-
                        UPDATE.-
!                       UNIT.-
!                       AMAT.-
!                       LMAT.-
                        CRAISE.-
                PLTDATA.-
!               MEAS.-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
!       UTILITY FUNCTIONS
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        VCRS,VDOT,VMAG,FQ
```

```
      SUBROUTINE MINV3(X,XINVRS)
C***********************************************************************
C         THIS SUBROUTINE FINDS THE INVERSE OF A 3X3 MATRIX
C***********************************************************************
      DIMENSION X(3,3),XINVRS(3,3),TCOFAC(3,3),EL(4)
      REAL*8 DET,SIGN,EL,TCOFAC,X,XINVRS
C***********************************************************************
C         FIND TRANSPOS  OF COFACTOR MATRIX AND DETERMINANT
C***********************************************************************
      SIGN = 1.0
      DO 400 I = 1,3
           DO 300 J = 1,3
                SIGN = -1.0*SIGN
                ICOUNT = 1
                DO 200 II = 1,3
                     DO 200 JJ = 1,3
                          IF (II.EQ.I.OR.JJ.EQ.J) GO TO 200
                          EL(ICOUNT) = X(II,JJ)
                          ICOUNT = ICOUNT + 1
  200                     CONTINUE
  300           TCOFAC(J,I) = SIGN*(EL(1)*EL(4)-EL(3)*EL(2))
  400      CONTINUE
C***********************************************************************
C         FIND DETERMINANT
C***********************************************************************
      DET = X(1,1)*TCOFAC(1,1) + X(2,1)*TCOFAC(1,2) + X(3,1)*TCOFAC(1,3)
C***********************************************************************
C         INVERSE = (TRANSPOSE OF COFACTOR MATRIX)/DETERMINANT
C***********************************************************************
      DO 500 I = 1,3
           DO 500 J = 1,3
  500           XINVRS(I,J) = TCOFAC(I,J)/DET
C
      RETURN
      END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	213	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	184	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD

ENTRY POINTS

Address	Type	Name
0-00000000		MINV3

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000068	R*8	DET	2-00000078	I*4	I	2-00000080	I*4	ICOUNT	2-00000084	I*4	II
2-0000007C	I*4	J	2-00000088	I*4	JJ	2-00000070	R*8	SIGN			

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000048	R*8	EL	32	(4)
2-00000000	R*8	TCOFAC	72	(3, 3)
AP-00000004@	R*8	X	72	(3, 3)
AP-00000008@	R*8	XINVRS	72	(3, 3)

A-336

LABELS

Address	Label	Address	Label	Address	Label	Address	Label
0-00000059	200	**	300	**	400	**	500

Total Space Allocated = 397 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

```
  Run Time:             1.08 seconds
  Elapsed Time:         11.19 seconds
  Page Faults:          325
  Dynamic Memory:       160 pages
```

```
0001        REAL*4 RSTART,RSTOP
0002        CHARACTER*62 MENU1(23)
0003        CHARACTER*40 YTITLE(26)
0004        CHARACTER*40 TITLES(26)
0005        LOGICAL*1 ATIT(1040),YTIT(1040),ATYPE
0006        EQUIVALENCE (ATIT(1),TITLES(1)),(YTITLE(1),YTIT(1))
0007        REAL*4       X(2000,23),TIME(2000),YARAY(2000),XFILL(13)
0008        DATA MENU1/
          * ' DISPLAY TYPES: ',' ',
          * ' 1  NAVIGATION ERROR   X POS     15  STAR TRACKER 2 RES. HORZ.',
          * ' 2  NAVIGATION ERROR   Y POS     16  STAR TRACKER 2 RES. VERT.',
          * ' 3  NAVIGATION ERROR   Z POS     17  L.M. TRACKER RESID. HORZ.',
          * ' 4  NAVIGATION ERROR   RSS POS   18  L.M. TRACKER RESID. VERT.',
          * ' 5  NAVIGATION ERROR   X VEL.    19  GPS X POSITION RESID.    ',
          * ' 6  NAVIGATION ERROR   Y VEL.    20  GPS Y POSITION RESID.    ',
          * ' 7  NAVIGATION ERROR   Z VEL.    21  GPS Z POSITION RESID.    ',
          * ' 8  NAVIGATION ERROR   RSS VEL   22  GPS RSS POSITION RESID.  ',
          * ' 9  ATTITUDE ERROR     ROLL      23  GPS X VELOCITY RESID.    ',
          * ' 10 ATTITUDE ERROR     PITCH     24  GPS Y VELOCITY RESID.    ',
          * ' 11 ATTITUDE ERROR     YAW       25  GPS Z VELOCITY RESID.    ',
          * ' 12 ATTITUDE ERROR     RSS       26  GPS RSS VELOCITY RESID.  ',
          * ' 13 STAR TRACKER 1 RES. HORZ     -1  EXIT                     ',
          * ' 14 STAR TRACKER 1 RES. VERT                                  ',
          * ' ',' ',' REFERENCE FRAMES TYPES :',' ',
          * ' 1  EARTH FIXED                  3  ORBIT              ',
          * ' 2  INERTIAL                     4  FLIGHT             ',
          * ' '/
0009        DATA YTITLE/' NAVIGATION ERROR  X POS (KM.) ',
          *             ' NAVIGATION ERROR  Y POS (KM.) ',
          *             ' NAVIGATION ERROR  Z POS (KM.) ',
          *             ' NAVIGATION ERROR  RSS POS. (KM.) ',
          *             ' NAVIGATION ERROR  X VEL. (KM./SEC.)',
          *             ' NAVIGATION ERROR  Y VEL. (KM./SEC.)',
          *             ' NAVIGATION ERROR  Z VEL. (KM./SEC.)',
          *             ' NAVIGATION ERROR  RSS VEL. (KM./SEC.)',
          *             ' ATTITUD  ERROR ROLL   (ARC SEC.)',
          *             ' ATTITUD  ERROR PITCH  (ARC SEC.)',
          *             ' ATTITUD  ERROR YAW    (ARC SEC.)',
          *             ' ATTITUD  ERROR RSS    (ARC SEC.)',
          *             ' S.T. 1 HORZ. RESIDUAL (ARC SEC.)',
          *             ' S.T. 1 VERT. RESIDUAL (ARC SEC.)',
          *             ' S.T. 2 HORZ. RESIDUAL (ARC SEC.)',
          *             ' S.T. 2 VERT. RESIDUAL (ARC SEC.)',
          *             ' L.M. TRACKER HORZ. RES. (ARC SEC.)',
          *             ' L.M. TRACKER VERT. RES. (ARC SEC.)',
          *             ' GPS X POS RESIDUAL (KM.) ',
          *             ' GPS Y POS RESIDUAL (KM.) ',
          *             ' GPS Z POS RESIDUAL (KM.) ',
          *             ' GPS RSS POS RESIDIDUAL (KM.)',
          *             ' GPS X V L RESIDUAL (KM./SEC.)',
          *             ' GPS Y V L RESIDUAL (KM./SEC.)',
          *             ' GPS Z V L RESIDUAL (KM./SEC.)',
          *             ' GPS RSS VEL RESID. (KM./SEC.)'/
0010        DATA TITLES/' NAVIGATION ERROR X POS.  VS.  TIME  ',
          *             ' NAVIGATION ERROR Y POS.  VS.  TIME    ',
          *             ' NAVIGATION ERROR Z POS.  VS.  TIME    ',
          *             ' NAVIGATION RSS POS. ERROR  VS.  TIME  ',
```

```
     *                  ' NAVIGATION ERROR X VEL.  VS.  TIME    ',
     *                  ' NAVIGATION ERROR Y VEL.  VS.  TIME    ',
     *                  ' NAVIGATION ERROR Z VEL.  VS.  TIME    ',
     *                  ' NAVIGATION RSS VEL. ERROR  VS.  TIME  ',
     *                  ' ATTITUDE ERROR ROLL  VS.  TIME        ',
     *                  ' ATTITUDE ERROR PITCH  VS.  TIME       ',
     *                  ' ATTITUDE ERROR YAW  VS.  TIME         ',
     *                  ' ATTITUDE ERROR RSS  VS.  TIME         ',
     *                  ' STAR TRACKER 1 HORZ. RES. VS. TIME    ',
     *                  ' STAR TRACKER 1 VERT. RES. VS. TIME    ',
     *                  ' STAR TRACKER 2 HORZ. RES. VS. TIME    ',
     *                  ' STAR TRACKER 2 VERT. RES. VS. TIME    ',
     *                  ' LANDMARK TRACKER HORZ. RES. VES. TIME ',
     *                  ' LANDMARK TRACKER VERT. RES. VS. TIME  ',
     *                  ' GPS X POSITION RESIDUAL VS. TIME      ',
     *                  ' GPS Y POSITION RESIDUAL VS. TIME      ',
     *                  ' GPS Z POSITION RESIDUAL VS. TIME      ',
     *                  ' GPS RSS POSITION RESIDUAL VS. TIME    ',
     *                  ' GPS X VELOCITY RESIDUAL VS. TIME      ',
     *                  ' GPS Y VELOCITY RESIDUAL VS. TIME      ',
     *                  ' GPS Z VELOCITY RESIDUAL VS. TIME      ',
     *                  ' GPS RSS VELOCITY RESIDUAL VS. TIME    '/
C*************************************************************
C--- DEFINITION OF X
C          X(I,1)      = TIME
C          X(I,2)      = X POS ERROR
C          X(I,3)      = Y POS ERROR
C          X(I,4)      = Z POS ERROR
C          X(I,5)      = X VEL ERROR
C          X(I,6)      = Y VEL ERROR
C          X(I,7)      = Z VEL ERROR
C          X(I,8-11)   = QUATERNION ERRORS
C          X(I,12)     = LM HORIZONTAL RESIDUAL
C          X(I,13)     = LM VERTICAL RESIDUAL
C          X(I,14)     = STAR TRACKER 1 RESIDUAL HORIZONTAL
C          X(I,15)     =  "      "    "     "     VERTICAL
C          X(I,16)     =  "      "    2     "     HORIZONTAL
C          X(I,17)     =  "      "    "     "     VERTICAL
C          X(I,18)     = GPS X POS RESIDUAL
C          X(I,19)     =  "  Y  "    "
C          X(I,20)     =  "  Z  "    "
C          X(I,21)     = GPS X VEL RESIDUAL
C          X(I,22)     =  "  Y  "    "
C          X(I,23)     =  "  Z  "    "
C*************************************************************
C
C--- START OF PLOTS
C
            OPEN(UNIT=2,NAME='PLOT.DAT',RECORDSIZE=660,TYPE='OLD')
            I=1
   10       CONTINUE
            READ(2,900,END=20)(X(I,J),J=1,23),(XFILL(K),K=1,10)
            I=I+1
            GO TO 10
   20       CONTINUE
            INUM=I-1
            WRITE(5,2)INUM
```

```
2       FORMAT(' TOTAL DATA VALUES READ IN WAS ',I8)
        WRITE(5,908)
        READ(5,909)ATYPE
30      CONTINUE
40      CONTINUE
        WRITE(5,901)(MENU1(I),I=1,23)
        WRITE(5,902)
        READ(5,903,ERR=40)ITYPE
        IF(ITYPE.EQ.-1)GO TO 890
        WRITE(5,904)
        READ(5,903,ERR=40)IRFNUM
        WRITE(5,905)X(1,1),X(INUM,1)
        READ(5,906,ERR=40)RSTART,RSTOP
        IF(RSTART.LT.RSTOP)GO TO 45
          WRITE(5,907)
          GO TO 40
45        CONTINUE
        IF(RSTOP.GT.X(INUM,1))RSTOP=X(INUM,1)
        IF(RSTART.LT.X(1,1))RSTART=X(1,1)
        IF(ATYPE.EQ.'H')GO TO 47
          CALL TK
          GO TO 48
47        CONTINUE
          CALL HP7221
          CALL QQHP(0.0,0.0,6)
48        CONTINUE
        CALL OPNPLT
        CALL XLABEL(' TIME  (SEC.) ',14)
        CALL YLABEL(YTIT((ITYPE-1)*40+1),40)
C
C--- FIND THE BOUNDS ON THE TIME
C
        DO 50 I=1,INUM-1
          IF(RSTART.GE.X(I,1).AND.RSTART.LE.X(I+1,1))GO TO 60
          WRITE(5,111)RSTART,X(I,1),X(I+1,1)
111       FORMAT(' RSTART,X(I),X(I+1) ',3F15.7)
50        CONTINUE
60        CONTINUE
        ITSTRT=I
        DO 70 I=1,INUM-1
          IF(RSTOP.GE.X(I,1).AND.RSTOP.LE.X(I+1,1))GO TO 80
70        CONTINUE
80        CONTINUE
        ITSTOP=I+1
        ITNUM=ITSTOP-ITSTRT+1
        DO 90 I=1,ITNUM
          TIME(I)=X(I+ITSTRT-1,1)
90        CONTINUE
        GO TO (100,100,100,200,100,100,100,200,300,300,300,400,500,500,
     *         500,500,500,500,500,500,500,600,500,500,500,600)ITYPE
100     CONTINUE
C
C--- NAVIGATIONAL ERROR TYPE OF PLOT
C
        ITYPE1=ITYPE
        IF(ITYPE1.LE.3)ITYPE1=ITYPE1+1
        DO 110 I=1,ITNUM
```

```
              YARAY(I)=X(I+ITSTRT-1,ITYPE1)
      110     CONTINUE
            GO TO 800
      200   CONTINUE
      C
      C--- RSS ERROR TYPE OF PLOT
      C
            ITYPE1=4
            IF(ITYPE.EQ.8)ITYPE1=7
            DO 210 I=1,ITNUM
              YARAY(I)=SQRT(X(I+ITSTRT-1,ITYPE1-2)**2+
     *                      X(I+ITSTRT-1,ITYPE1-1)**2+
     *                      X(I+ITSTRT-1,ITYPE1)**2)
      210     CONTINUE
            GO TO 800
      300   CONTINUE
      C
      C--- ATTITUDE ERROR
      C
            DO 310 I=1,ITNUM
              YARAY(I)=(X(I+ITSTRT-1,ITYPE)*2.)/.4848E-5
      310     CONTINUE
            GO TO 800
      400   CONTINUE
      C
      C--- RSS ATTITUDE ERROR
      C
            DO 410 I=1,ITNUM
              YARAY(I)=(2*SQRT(X(I+ITSTRT-1,9)**2+
     *          X(I+ITSTRT-1,10)**2+X(ITSTRT+I-1,11)**2))/.4848E-5
      410     CONTINUE
            GO TO 800
      C
      C--- RESIDUAL TYPE OF PLOT
      C
      500   CONTINUE
            ITYPE1=ITYPE-1
            IF(ITYPE.GE.23)ITYPE1=ITYPE-2
            DO 510 I=1,ITNUM
              YARAY(I)=X(I+ITSTRT-1,ITYPE1-1)
      510     CONTINUE
            GO TO 800
      C
      C--- RSS TYPE FOR RESIDUALS
      C
      600   CONTINUE
            ITYPE1=18
            IF(ITYPE.EQ.25)ITYPE1=21
            DO 610 I=1,ITNUM
              YARAY(I)=2.*SQRT(X(I+ITSTRT-1,ITYPE1)**2+
     *                         X(I+ITSTRT-1,ITYPE1+1)**2+
     *                         X(I+ITSTRT-1,ITYPE1+2)**2)
      610     CONTINUE
            GO TO 800
      800   CONTINUE
            CALL LINPLT(TIME,YARAY,ITNUM)
            IF(ATYPE.EQ.'H')CALL QQHP(1.0,0.0,6)
```

```
              CALL GRID(2,2)
              IF(ATYPE.EQ.'H')CALL QQMP(0.,0.,6)
              CALL HEADIN(ATIT((ITYPE-1)*40+1),-40,-2,1)
              CALL ENDPL(0)
              GO TO 40
      890     CONTINUE
      900     FORMAT(33F20.10)
      901     FORMAT(A60)
      902     FORMAT(' INPUT DISPLAY TYPE NUMBER    : ',$)
      903     FORMAT(I2)
      904     FORMAT(' INPUT REFERENCE FRAME NUMBER : ',$)
      905     FORMAT(' INPUT START,STOP TIME FOR PLOT : ',/,
             *     ' MIN MAX TIMES ARE ',2F10.3,5X,$)
      906     FORMAT(2F15.6)
      907     FORMAT(' BAD START,STOP TIMES WERE INPUT ')
      908     FORMAT(' INPUT OUTPUT DEVICE (T OR H) : ',$)
      909     FORMAT(A1)
              END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	1516	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	360	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	203772	PIC CON REL LCL NOSHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		PLOTSMAIN

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00031B26	L*1	ATYPE	2-00031B30	I*4	I	2-00031B3C	I*4	INUM	2-00031B44	I*4	IRFNUM
2-00031B50	I*4	ITNUM	2-00031B4C	I*4	ITSTOP	2-00031B48	I*4	ITSTRT	2-00031B40	I*4	ITYPE
2-00031B54	I*4	ITYPE1	2-00031B34	I*4	J	2-00031B38	I*4	K	2-00031B28	R*4	RSTART
2-00031B2C	R*4	RSTOP									

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000410	L*1	ATIT	1040	(1040)
2-00031594	CHAR	MENU1	1426	(23)
2-0002D6E0	R*4	TIME	8000	(2000)
2-00000410	CHAR	TITLES	1040	(26)
2-00000820	R*4	X	184000	(2000, 23)

```
2-00031560  R*4   XFILL            52  (13)
2-0002F620  R*4   YARAY          8000  (2000)
2-00000000  L*1   YTIT           1040  (1040)
2-00000000  CHAR  YTITLE         1040  (26)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000000	2'	0-00000015	10	0-00000072	20	**	30	0-000000CE	40	0-00000208	45
0-0000023C	47	0-00000248	48	**	50	0-000002DF	60	**	70	0-00000313	80
**	90	0-00000385	100	**	110	1-00000024	111'	0-000003C6	200	**	210
0-0000043A	300	**	310	0-00000476	400	**	410	0-000004D6	500	**	510
0-0000051C	600	**	610	0-00000590	800	0-000005E8	890	1-00000040	900'	1-00000046	901'
1-00000049	902'	1-00000060	903'	1-00000070	904'	1-00000094	905'	1-000000D7	906'	1-000000DD	907'
1-00000101	908'	1-00000125	909'								

FUNCTIONS AND SUBROUTINES REFERENCED

```
ENDPL     FOR$OPEN   GRID     HEADIN   HP7221   LINPLT   MTH$SQRT   OPNPLT
QQHP      TK         XLABEL   YLABEL
```

Total Space Allocated = 205648 Bytes

COMMAND QUALIFIERS

```
FORTRAN /CO:40 PLOT/LIS

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=40
```

A-343

COMPILATION STATISTICS

```
Run Time:          4.67 seconds
Elapsed Time:      19.17 seconds
Page Faults:       294
Dynamic Memory:    82 pages
```

```
0001          SUBROUTINE PM AS(PU,U,PDH,PDV,IFLAG)
        C
        C**************************************************************
        C
        C     SUBROUTINE PM AS COMPUTES THE PARTIALS OF THE MEASUREMENT
        C     WITH RESPECT TO THE ESTIMATED STATE USING THE PARTIAL OF
        C     THE UNIT VECTOR (PU) WRT THE ESTIMATED STATE AND THE
        C     UNIT VECTOR ITSELF.
        C
        C     INPUT PARAMET RS
        C             PU       = THE PARTIALS OF THE UNIT VECTOR WRT THE
        C                             ESTIMATED STATE
        C             U        = THE MEASUREMENT UNIT VECTOR ITSELF
        C     OUTPUT PARAMETERS
        C             PDH      = THE PARTIALS OF THE HORIZONTAL DIFLECTION
        C             PDV      = THE PARTIALS OF THE VERTICAL DIFLECTION
        C     CALLED BY HSTAR AND HLMT
        C
        C**************************************************************
        C
0002          DIMENSION PU(3),U(3)
0003          REAL*8 PDH,PDV,PU,U,A,B,C
0004          IF(IFLAG .GT. 1) GO TO 100
        C
        C**************************************************************
        C     COMPUTE PARTIALS OF MEASUREMENT VECTOR (APPROXIMATE)
        C**************************************************************
        C
0005          PDH=PU(2)
0006          PDV=PU(1)
0007          RETURN
        C
        C**************************************************************
        C     COMPUTE EXACT PARTIALS OF MEASUREMENT
        C**************************************************************
        C
0008    100   A=1./SQRT(1.-U(1)*U(1))
0009          PDV=A*PU(1)
0010          B=COS(ASIN(U(1)))
0011          C=1./SQRT(1.-(U(2)/B)**2)
0012          PDH=C*(B*PU(2)+U(2)*U(1)*A*PU(1))/(B*B)
0013          RETURN
0014          END
```

```
PROGRAM SECTIONS

    Name                                       Bytes   Attributes

  0 $CODE                                        163   PIC CON REL LCL   SHR   EXE   RD NOWRT LONG
  2 $LOCAL                                        64   PIC CON REL LCL NOSHR NOEXE  RD   WRT QUAD

ENTRY POINTS

    Address   Type  Name

  0-00000000        PMEAS

VARIABLES

    Address   Type  Name            Address   Type  Name            Address   Type  Name            Address   Type  Name

  2-00000000  R*8   A             2-00000008  R*8   B             2-00000010  R*8   C             AP-00000014@ I*4  IFLAG
 AP-0000000C@ R*8   PDH          AP-00000010@ R*8   PDV

ARRAYS

    Address   Type  Name                       Bytes  Dimensions

 AP-00000004@ R*8   PU                            24  (3)
 AP-00000008@ R*8   U                             24  (3)

LABELS

    Address   Label

  0-00000034  100

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DASIN     MTH$DCOS     MTH$DSQRT

Total Space Allocated = 227 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:            0.86 seconds
  Elapsed Time:        18.87 seconds
  Page Faults:         323
  Dynamic Memory:      160 pages
```

```
0001          SUBROUTINE QMULT (A,B,C)
        C*****************************************************************
        C THIS SUBROUTINE DOES A QUATERNION MULTIPLY AND RETURNS ANSWER IN C
        C*****************************************************************
0002          DIMENSION A(4),B(4),C(4),D(3)
0003          REAL*8 A,B,C,D
        C
0004          D(1) = A(1)*B(1) - A(2)*B(2) - A(3)*B(3) - A(4)*B(4)
0005          D(2) = A(1)*B(2) + A(2)*B(1) + A(3)*B(4) - A(4)*B(3)
0006          D(3) = A(1)*B(3) - A(2)*B(4) + A(3)*B(1) + A(4)*B(2)
0007          C(4) = A(1)*B(4) + A(2)*B(3) - A(3)*B(2) + A(4)*B(1)
0008          C(1) = D(1)
0009          C(2) = D(2)
0010          C(3) = D(3)
        C
0011          RETURN
0012          END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	210	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	84	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD

ENTRY POINTS

Address	Type	Name
0-00000000		QMULT

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	A	32	(4)
AP-00000008@	R*8	B	32	(4)
AP-0000000C@	R*8	C	32	(4)
2-00000000	R*8	D	24	(3)

Total Space Allocated = 294 Bytes

COMMAND QUALIFIERS

FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

/CHECK=(NOBOUNDS,OVERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time:	1.07 seconds
Elapsed Time:	14.16 seconds
Page Faults:	293
Dynamic Memory:	160 pages

```
0001            SUBROUTINE RDDTA(T,THET,AL)
        C
        C***********************************************************
        C       SUBROUTINE RDDTA READS THE VALUE OF THET AND AL FROM
        C       A DATA TABLE AT THE TIME CLOSEST TO TIME T.
        C
        C       INPUT DATA
        C               T       = TIME OF MEASUREMENT
        C
        C       OUTPUT DATA
        C               THET    = LOOK ANGLE TO LANDMARK (RAD)
        C               AL      = ALTITUDE OF LANDMARK ABOVE EARTH MEAN
        C                               SURFACE (KM)
        C
        C       CALLED BY RDGCP
        C
        C       THET IS INITIALLY A FLAG
        C               THET .GT. 90. DEG       PICK RANDOM THET AND AL
        C               THET .LT. 90. DEG       KEEP THET, SET AL=0.
        C
        C
        C***********************************************************
        C
0002            INCLUDE 'MODE.COM'
0003    *       COMMON /MODE/ MODE(10)
        * C
        * C             MODE(1) = LANDMARK TRACKER SWEEP MODE
        * C                             0 = RANDOM
        * C                             1 = FIXED AT INPUT THET
        * C                             2 = NO DEFAULT TO STAR TRACKER
        * C             MODE(2) = CLOUD SELECTION MODE
        * C                             0 = RANDOM CLOUD DENSITIES BASED
        * C                                     ON INPUT TABLES CLOTBL
        * C                             1 = FIXED DENSITY AT NO CLOUDS
        * C                             2 = NO CLOUDS WITH 100% CLOUD
        * C                                     COVER FOR A SPECIFIED
        * C                                     PERIOD (CLOTBL(11,12))
        * C             MODE(3-10) NOT SPECIFIED AT PRESENT
        * C
0004            DATA I1,I2,IFLAG/12345,54321,0/
0005            REAL*8 AL,T,THET,RAN,GAUSS,X,THETP
0006            IF(IFLAG .GT. 0) GO TO 10
0007            IFLAG=1
0008            THETP=THET*3.1415926536/180.
0009     10     GO TO (20,20),MODE(1)
0010            THET=(RAN(I1,I2)-.5)*2.*THETP
0011            X=GAUSS(0.,.5)
0012            AL=ABS(X)
0013            GO TO 100
0014     20     THET=THETP
0015            AL=0.
0016    100     RETURN
0017            END
```

```
PROGRAM SECTIONS

    Name                          Bytes   Attributes

  0 $CODE                           110   PIC CON REL LCL   SHR    EXE   RD NOWRT LONG
  1 $PDATA                            8   PIC CON REL LCL   SHR NOEXE   RD NOWRT LONG
  2 $LOCAL                           52   PIC CON REL LCL NOSHR NOEXE   RD   WRT QUAD
  3 MODE                             40   PIC OVR REL GBL   SHR NOEXE   RD   WRT LONG

ENTRY POINTS

    Address   Type  Name

  0-00000000        RDDTA

VARIABLES

    Address   Type  Name            Address   Type  Name            Address   Type  Name            Address   Type  Name

  AP-0000000C@ R*8  AL             2-00000010  I*4  I1             2-00000014  I*4  I2             2-00000018  I*4  IFLAG
  AP-00000004@ R*8  T            AP-00000008@ R*8  THET           2-00000008  R*8  THETP          2-00000000  R*8  X

ARRAYS

    Address   Type  Name                 Bytes  Dimensions

  3-00000000  I*4   MODE                    40  (10)

LABELS

    Address    Label       Address    Label       Address    Label

  0-0000002C   10        0-00000065   20        0-0000008D   100

FUNCTIONS AND SUBROUTINES REFERENCED

FOR$IRAN      GAUSS

Total Space Allocated = 210 Bytes

COMMAND QUALIFIERS

  FORTRAN /L RDDTA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:            0.83 seconds
  Elapsed Time:        7.44 seconds
  Page Faults:         149
  Dynamic Memory:      39 pages
```

```
0001            SUBROUTINE RDGCP
          C*****************************************************************
          C
          C     SUBROUTINE RDGCP READS FROM THE DATA BASE LONGITUDE
          C     LATITUDE, AND ALTITUDE FOR THE INTENDED LANDMARK
          C
          C     INPUT PARAMETERS
          C            THET    = LOOK ANGLE ABOUT LMT X-AXIS (RAD)
          C            AL      = LANDMARK ALTITUDE ABOVE MEAN (KM)
          C     OUTPUT PARAMETERS
          C            LON     = LANDMARK LONGITUDE (DEG)
          C            LAT     =    "     LATITUDE (DEG)
          C            AL      =    "     ALTITUDE ABOVE MEAN (KM)
          C
          C     CALLED BY MEASURE
          C
          C*****************************************************************
          C
0002            REAL*8 B,RE,TO,TIL,LE,LI,M,TBI,TEI,TIB,TIE,TLL,UL,
               *       ULI,ULL,VMAG,DOWN,THETP,STHETP,VCRS,UCD
                INCLUDE 'LMTPAR.COM'
0003            COMMON /LMTPAR/ AL,LON,LAT,TBNL(3,3),TNL(3,3),BL(2),SL(2),
0004     *     *       BKL(2),SKL(2),TNLK(3,3),TIEO(3,3),SIGGCP,THET
         *      REAL*8 AL,TBNL,TNL,BL,SL,BKL,SKL,TNLK,TIEO,SIGGCP,LAT,LON,
0005     *     *       THET
         *
         * C
         * C     LANDMARK TRACKER PARAMETERS
         * C            AL      = ALTITUDE OF LANDMARK (KM)
         * C            LON     = LONGITUDE OF LANDMARK (DEG)
         * C            LAT     = LATITUDE OF LANDMARK (DEG)
         * C            TBNL    = ORIENTATION ARRAY FOR LANDMARK TRACKER
         * C                            NOMINAL TO BODY
         * C            TNL     = MISALIGNMENT ARRAY - ACTUAL
         * C                            TRACKER TO NOMINAL
         * C            BL      = BIAS - ACTUAL (RAD)
         * C            SL      = NOISE STANDARD DEVIATION - ACTUAL (RAD)
         * C            BKL     = BIAS - KNOWLEDGE (RAD)
         * C            THET    = LOOK ANGLE (RAD)
         * C            SKL     = NOISE STANDARD DEVIATION -KNOWLEDGE (RAD)
         * C            TIEO    = INITIAL EARTH FIXED TO INERTIAL
         * C                            TRANSFORMATION
         * C            TNLK    = MISALIGNMENT ARRAY KNOWLEDGE
         * C                            TRACKER TO NOMINAL
         * C            SIGGCP  = POSITION UNCERTAINTY DUE TO CLOUDS
         * C
         * C
                INCLUDE 'ENVIR.COM'
0006     *      COMMON /ENVIR/ STATE(10),PROFILE(10,4),INIT
0007     *      REAL*8 STATE,PROFILE
0008     * C
         * C            REAL WORLD STATE PARAMETERS
         * C
         * C                 STATE     STATE VALUES; X,Y,Z,XD,YD,ZD,E0,E1,E2,E3
         * C                 PROFILE   ATTITUDE PROFILE-TIME (SEC) VS
         * C                           INERTIAL ANGULAR RATES (RAD/SEC)
         * C                 INIT      INTEGRATION INITIALIZATION KEY (-1)
         * C
         * C
```

```
02300   0009            INCLUDE 'ARRAYS.COM'
02400          * C
02400   0010   *        COMMON /ARRAYS/ T1(3),T2(3),T3(3),T4(10),T11(3,3),T33(3,3)
02400          *        .         ,T44(4,4),T66(6,6),T77(6,6),T5(4),T6(4),T7(4)
02400   0011   *        REAL*8 T1,T2,T3,T4,T11,T33,T44,T66,T77,T5,T6,T7
02400          * C
02400          * C              THESE ARE TEMPORARY STORAGE ARRAYS FOR USE BY ALL MODULES
02400          * C
02400          * C                    T1 - T4      SINGLE DIMENSION ARRAYS
02400          * C                    T11 - T77    DUAL DIMENSIONED ARRAYS
02400          * C                    T11          DUAL ARRAY; OFF DIAGIONAL SET TO ZERO
02400          * C
02400   0012            INCLUDE 'TARGETS.COM'
02500   0013   *        COMMON /TARGETS/ MTYPE,IS,NS,JFLAG,MCODE,PI,TPI
02500   0014   *        LOGICAL JFLAG
02500   0015   *        REAL*8 PI,TPI
02500          * C
02500          * C              MEASUREMENT SPECIFICATIONS
02500          * C
02500          * C                    MTYPE      MEASUREMENT TYPE
02500          * C                    JFLAG      SET FOR STAR OBSTRUCTION
02500          * C                    MCODE       "    "  MEASUREMENT PROCESSING
02500          * C                    PI         PI
02500          * C                    TPI        2*PI
02500          * C
02500   0016            INCLUDE 'TIME.COM'
00100          * C
00200   0017   *        COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
00300          *        .         ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
00400   0018   *        REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
00500          *        .     TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
00600          * C
00700          * C              THESE ARE THE TIME REFERENCE FRAMES
00800          * C
00900          * C                    TIME       ATOMIC TIME SINCE INITIALIZATION (SEC)
01000          * C                    TNEXT      TIME FOR NEXT POSITION INTEGRATION (SEC)
01100          * C                    TSTOP      RUN TERMINATION TIME (SEC)
01200          * C                    TIA        ATTITUDE INTEGRATION TIME (SEC)
01300          * C                    DEL         "            "        STEP SIZE (SEC)
01400          * C                    TIN        POSITION INTEGRATION TIME (SEC)
01500          * C                    DTN         "           "         STEP SIZE (SEC)
01600          * C                    DATEO      DATE OF FLIGHT EPOCH (JD)
01700          * C                    DATER      DATE OF 1950 EPOCH (JD)
01800          * C                    TZERO      START TIME IN SECS. SINCE DATEO
01900          * C                    TSLEW      TIME NEEDED TO SLEW AND ACQUIRE (SEC)
02000          * C                    TIS        REAL WORLD REFERENCE TIME (SEC)
02100          * C                    TISN       TIME FOR NEXT RW POSITION INTEGRATION (SEC)
02200          * C                    DTA        USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
02300          * C                               TOO LARGE AT MEASUREMENT TIME
02400          * C                    TPRINT     TIME FOR PRINT (SEC)
02500          * C                    DTPRINT    INCREMENT ON TPRINT (SEC)
02600          * C
02600   0019            INCLUDE 'NSTATE.COM'
02700          * C
02700   0020   *        COMMON /NSTATE/ XD(6),X(6),RADM,RADE
02700   0021   *        REAL*8 XD,X,RADM,RADE
02700          * C
```

```
              * C              POSITION STATE AND CONSIDERED PARAMETERS
              * C
              * C                   XD        STATE DERIVATIVES (KM/SEC AND KM/SEC/SEC)
              * C                   X         STATE POSITION PARAMETERS (KM AND KM/SEC)
              * C                   RADM      RADIUS OF THE MOON (KM)
              * C                   RADE      EARTH DETECTABLE RADIUS (KM)
              * C
              * C
0022                  DIMENSION UL(3),TLL(3,3),ULL(3),TBI(3,3),TIB(3,3),ULI(3),
                    .          M(3),LI(3),TIE(3,3),TEI(3,3),LE(3),TIL(3,3),
                    .          DOWN(3),UCD(3)
0023                  EQUIVALENCE (RE,RADE)
                C
                C**********************************************************
                C        GENERATE UNIT VECTOR ALONG LMT BORESIGHT
                C**********************************************************
                C
0024                  UL(1)=0.
0025                  UL(2)=0.
0026                  UL(3)=1.
                C
                C**********************************************************
                C        READ A LOOK ANGLE TO AND AN ALTITUDE OF THE LANDMARK
                C        FROM THE DATA FILE
                C        LOOK ANGLE IS RELATIVE TO BODY AXES ABOUT THE
                C             X-BODY AXIS
                C**********************************************************
                C
0027                  CALL RDDTA(TIME,THET,AL)
                C
                C**********************************************************
                C        TRANSFORM UNIT VECTOR BY LOOK ANGLE
                C**********************************************************
                C
0028                  DO 10 I=1,3
0029                  DO 10 J=1,3
0030            10    TLL(I,J)=0.
0031                  TLL(1,1)=1.
0032                  TLL(2,2)=COS(THET)
0033                  TLL(3,3)=TLL(2,2)
0034                  TLL(2,3)=-SIN(THET)
0035                  TLL(3,2)=-TLL(2,3)
0036                  CALL MATAB(TLL,UL,ULL,3,3,1)
                C
                C**********************************************************
                C        TRANSFORM UNIT VECTOR TO INERTIAL COORDINATES
                C**********************************************************
                C
0037                  CALL AMAT(STATE(7),TBI)
0038                  CALL MINV3(TBI,TIB)
0039                  CALL MATAB(TBNL,TNL,T33,3,3,3)
0040                  CALL MATAB(TIB,T33,TIL,3,3,3)
0041                  CALL MATAB(TIL,ULL,ULI,3,3,1)
                C
                C**********************************************************
                C        DETERMINE ANGLE OF UNIT VECTOR ALONG LANDMARK
                C        DIRECTION RELATIVE TO LOCAL DOWN
```

```
            C**************************************************************
            C
0042                DO 15 I=1,3
0043        15      DOWN(I)=-STAT (I)
0044                CALL VCRS(ULI,DOWN,UCD)
0045                STHETP=VMAG(UCD,3)/VMAG(DOWN,3)
0046                THETP=ASIN(STHETP)
            C
            C**************************************************************
            C       FIND LENGTH OF VECTOR FROM S/C TO LM
            C**************************************************************
            C
0047                B=SQRT(VMAG(STATE,3)**2+(RE+AL)**2-2.*VMAG(STATE,3)*
                .             (RE+AL)*COS(-THETP+ASIN(VMAG(STATE,3)*
                .                         SIN(THETP)/(RE+AL))))
            C
            C**************************************************************
            C       ESTABLISH VECTOR
            C**************************************************************
            C
0048                DO 20 I=1,3
0049        20      M(I)=B*ULI(I)
            C
            C**************************************************************
            C       COMPUTE LM POSITION VEC R IN INERTIAL SPACE
            C**************************************************************
            C
0050                DO 30 I=1,3
0051        30      LI(I)=STATE(I)+M(I)
            C
            C**************************************************************
            C       TRANSFORM TO ARTH FIFED COORDINATES
            C**************************************************************
            C
0052                CALL WET(TIEO,TIE)
0053                CALL MINV3(TI_,TEI)
0054                CALL MATAB(TEI,LI,LE,3,3,1)
            C
            C**************************************************************
            C       SOLVE FOR LATITUDE AND LONGIT DE (DEG)
            C**************************************************************
            C
0055                LAT=ASIN(LE(3)/(RE+AL))
0056                LON=180.*ASIN(LE(2)/((RE+AL)*COS(LAT)))/PI
0057                LAT=LAT*180./PI
0058                RETURN
0059                END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes								
0	$CODE	505	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
1	$PDATA	8	PIC	CON	REL	LCL	SHR	NOEXE	RD	NOWRT	LONG
2	$LOCAL	920	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	QUAD
3	LMTPAR	392	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
4	ENVIR	404	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
5	ARRAYS	1096	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
6	TARGETS	36	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
7	TIME	136	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG
8	NSTATE	112	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		RDGCP

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
3-00000000	R*8	AL	2-00000270	R*8	B	7-00000038	R*8	DATE0	7-00000070	R*8	DATER
7-00000020	R*8	DEL	7-00000068	R*8	DTA	7-00000030	R*8	DTN	7-00000080	R*8	DTPRINT
2-00000290	I*4	I	4-00000190	I*4	INIT	6-00000004	I*4	IS	2-00000294	I*4	J
6-0000000C	L*4	JFLAG	3-00000010	R*8	LAT	3-00000008	R*8	LON	6-00000010	I*4	MCODE
6-00000000	I*4	MTYPE	6-00000008	I*4	NS	6-00000014	R*8	PI	8-00000068	R*8	RADE
8-00000060	R*8	RADM	8-00000068	R*8	RE	3-00000178	R*8	SIGGCP	2-00000288	R*8	STHETP
2-00000278	R*8	T0	3-00000180	R*8	THET	2-00000280	R*8	THETP	7-00000018	R*8	TIA
7-00000000	R*8	TIME	7-00000028	R*8	TIN	7-00000058	R*8	TIS	7-00000060	R*8	TISN
7-00000048	R*8	TMEAS	7-00000008	R*8	TNEXT	6-0000001C	R*P	TPI	7-00000078	R*8	TPRINT
7-00000050	R*8	TRACK	7-00000010	R*8	TSTOP	7-00000040	R*8	TZERO			

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-000000C8	R*8	BKL	16	(2)
3-000000A8	R*8	BL	16	(2)
2-00000240	R*8	DOWN	24	(3)
2-00000048	R*8	LE	24	(3)
2-00000060	R*8	LI	24	(3)
2-00000078	R*8	M	24	(3)
4-00000050	R*8	PROFILE	320	(10, 4)
3-000000D6	R*8	SKL	16	(2)
3-00000088	R*8	SL	16	(2)
4-00000000	R*8	STATE	80	(10)
5-00000000	R*8	T1	24	(3)
5-00000098	R*8	T11	72	(3, 3)
5-00000018	R*8	T2	24	(3)
5-00000030	R*8	T3	24	(3)
5-00[illegible]0	R*8	T33	72	(3, 3)

```
5-00000048   R*8   T4          80   (10)
5-00000128   R*8   T44        125   (4, 4)
5-000003E8   R*8   T5          32   (4)
5-00000408   R*8   T6          32   (4)
5-000001A8   R*8   T66        288   (6, 6)
5-00000428   R*8   T7          32   (4)
5-000002C8   R*8   T77        288   (6, 6)
2-00000090   R*8   TBI         72   (3, 3)
3-00000018   R*8   TBNL        72   (3, 3)
2-000000D8   R*8   TEI         72   (3, 3)
2-00000120   R*8   TIB         72   (3, 3)
2-00000168   R*8   TIE         72   (3, 3)
3-00000130   R*8   TIEO        72   (3, 3)
2-00000000   R*8   TIL         72   (3, 3)
2-000001B0   R*8   TLL         72   (3, 3)
3-00000060   R*8   TNL         72   (3, 3)
3-000000E8   R*8   TNLK        72   (3, 3)
2-00000258   R*8   UCD         24   (3)
2-000001F8   R*8   UL          24   (3)
2-00000210   R*8   ULI         24   (3)
2-00000228   R*8   ULL         24   (3)
8-00000030   R*8   X           48   (6)
8-00000000   R*8   XD          48   (6)
```

LABELS

```
Address   Label     Address   Label     Address   Label     Address   Label

  **       10         **       15         **       20         **       30
```

FUNCTIONS AND SUBROUTINES REFERENCED

```
AMAT      MATAB     MINV3     MTH$DASIN   MTH$DCOS   MTH$DSIN   MTH$DSQRT   RDDTA
VCRS      VMAG      WET
```

Total Space Allocated = 3609 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           2.57 seconds
  Elapsed Time:       31.83 seconds
  Page Faults:        395
  Dynamic Memory:     160 pages
```

```
              SUBROUTINE TRANFM(XTRANS,XTRANP)
        C
        C--- THIS ROUTINE CREATES THE TRANSFORMATION MATRICES FOR THE
        C--- TRANSFORMATION FROM INERTIAL TO FLIGHT COORDINATES.
        C
              INCLUDE 'ARRAY.COM'
      * C     COMMON BLOCK USED FOR PLOTTING ROUTINE
      * C
      *       COMMON /ARRAY/ X(3000,39,2), INUM
      *       REAL*4 X,X1(3000,39),X2(3000,39)
      *       INTEGER*2 INUM
      *       EQUIVALENCE (X1(1,1),X(1,1,1)),(X2(1,1),X(1,1,2))
      * C
              REAL*8 XTRANS(3,3,3000),XTRANP(3,3,3000)
              REAL*8 UXF(3,3000),UYF(3,3000),UZF(3,3000),TUYF(3,3000)
              REAL*8 XTEMP1(3),XTEMP2(3)
        C
        C--- CREATE Uzf
        C
              DO 10 I=1,INUM
                XTEMP1(1)=X(I,34,1)
                XTEMP1(2)=X(I,35,1)
                XTEMP1(3)=X(I,36,1)
                CALL UNIT(XTEMP1(1),UZF(1,I),3)
        10      CONTINUE
        C
        C--- NEGATE ALL ELEMENTS OF UZF
        C
              DO 15 I=1,INUM
                UZF(1,I)=-UZF(1,I)
                UZF(2,I)=-UZF(2,I)
                UZF(3,I)=-UZF(3,I)
        15      CONTINUE
        C
        C--- CREATE Uyf
        C
              DO 20 I=1,INUM
                XTEMP2(1)=X(I,37,1)
                XTEMP2(2)=X(I,38,1)
                XTEMP2(3)=X(I,39,1)
                CALL VCRS(XTEMP2(1),XTEMP1(1),TUYF(1,I))
                CALL UNIT(TUYF(1,I),UYF(1,I),3)
        20      CONTINUE
        C
        C--- CREATE Uxf
        C
              DO 30 I=1,INUM
                CALL VCRS(UYF(1,I),UZF(1,I),UXF(1,I))
        30      CONTINUE
        C
        C--- MERGE INTO XTRANS
        C
              DO 40 I=1,INUM
                DO 35 J=1,3
                  XTRANS(1,J,I)=UXF(J,I)
                  XTRANS(2,J,I)=UYF(J,I)
                  XTRANS(3,J,I)=UZF(J,I)
```

```
35          CONTINUE
40        CONTINUE
C
C--- CREATE THE INVERSE
C
          DO 50 I=1,INUM
            CALL MINV3(XTRANS(1,1,I),XTRANS(1,1,I))
50          CONTINUE
C
C
C--- CREATE THE TRANSPOSE
C
          DO 70 I=1,INUM
            DO 60 J=1,3
              XTRANP(1,J,I)=XTRANS(J,1,I)
              XTRANP(2,J,I)=XTRANS(J,2,I)
              XTRANP(3,J,I)=XTRANS(J,3,I)
60            CONTINUE
70          CONTINUE
          RETURN
          END
```


PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	527	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	4	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	288176	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 ARRAY	936002	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		TRANFM

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00046530	I*4	I	3-000E4840	I*2	INUM	2-00046534	I*4	J

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-000348C0	R*8	TUYF	72000	(3, 3000)
2-00000000	R*8	UXF	72000	(3, 3000)
2-00011940	R*8	UYF	72000	(3, 3000)
2-00023280	R*8	UZF	72000	(3, 3000)

```
 3-00000000  R*4  X            936000  (3000, 39, 2)
 3-00000000  R*4  X1           468000  (3000, 39)
 3-00072420  R*4  X2           468000  (3000, 39)
 2-00046500  R*8  XTEMP1           24  (3)
 2-00046518  R*8  XTEMP2           24  (3)
AP-00000008@ R*8  XTRANP       216000  (3, 3, 3000)
AP-00000004@ R*8  XTRANS       216000  (3, 3, 3000)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
**	10	**	15	**	20	**	30	**	35	**	40
**	50	**	60	**	70						

FUNCTIONS AND SUBROUTINES REFERENCED

MINV3 UNIT VCRS

Total Space Allocated = 1224709 Bytes

COMMAND QUALIFIERS

```
  FORTRAN TRANFM/LIS

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOMACHINE_CODE /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          2.15 seconds
  Elapsed Time:      4.93 seconds
  Page Faults:       187
  Dynamic Memory:    59 pages
```

```
        SUBROUTINE UNIT(R,V,N)
C*************************************************************************
C                       UNITIZE THE VECTOR V(3)
C*************************************************************************
        INCLUDE 'DEBUG.COM'
        *       COMMON /DEBUG/ IENTER,IDEBUG
        * C
        * C               USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
        * C
        * C                   IENTER    IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
        * C                   IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
        * C
          C      DIMENSION R(N),V(N)
                  DIMENSION R(22),V(22)
                  REAL*8 D,R,V,VMAG
                 D = VMAG(R,N)
                 DO 10 I=1,N
            10   V(I) = R(I)/D
                 RETURN
                 END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	87	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	64	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		UNIT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000000	R*8	D	2-00000008	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER
AP-0000000C@	I*4	N									

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	R	176	(22)
AP-00000008@	R*8	V	176	(22)

LABELS

```
    Address   Label

      **      10
```

FUNCTIONS AND SUBROUTINES REFERENCED

VMAG

Total Space Allocated = 159 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:          0.59 seconds
  Elapsed Time:      8.42 seconds
  Page Faults:       333
  Dynamic Memory:    160 pages
```

```
0001              SUBROUTINE VCRS(A,B,C)
0002              INCLUDE 'DEBUG.COM'
00100    0003     *       COMMON /DEBUG/ IENTER,IDEBUG
00200             * C
00300             * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400             * C
00500             * C                     IENTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600             * C                     IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
00700             * C
         0004             DIMENSION A(3),B(3),C(3)
         0005             REAL*8 A,B,C
                    C
                    C          VECTOR CROSS PRODUCT  (A) X (B) = (C)
                    C
         0006             C(1) = A(2)*B(3)-A(3)*B(2)
         0007             C(2) = A(3)*B(1)-A(1)*B(3)
         0008             C(3) = A(1)*B(2)-A(2)*B(1)
         0009             RETURN
         0010             END
```

PROGRAM SECTIONS

Name	Bytes	Attributes								
0 $CODE	106	PIC	CON	REL	LCL	SHR	EXE	RD	NOWRT	LONG
2 $LOCAL	60	PIC	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	LONG
3 DEBUG	8	PIC	OVR	REL	GBL	SHR	NOEXE	RD	WRT	LONG

ENTRY POINTS

Address	Type	Name
0-00000000		VCRS

VARIABLES

Address	Type	Name	Address	Type	Name
3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	A	24	(3)
AP-00000008@	R*8	B	24	(3)
AP-0000000C@	R*8	C	24	(3)

Total Space Allocated = 174 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:           0.63 seconds
  Elapsed Time:       10.88 seconds
  Page Faults:        165
  Dynamic Memory:     160 pages
```

```
0001            FUNCTION VDOT(A,B,N)
          C*********************************************************************
          C        VECTOR DOT PRODUCT (A) DOT (B) = VDOT
          C*********************************************************************
0002            INCLUDE 'DEBUG.COM'
00100     0003  *        COMMON /DEBUG/ IENTER,IDEBUG
00200           * C
00300           * C              USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400           * C
00500           * C                   I NTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600           * C                   IDEBUG     0-10, HIGHER NUMBER MEANS MORE PRINT
00700           * C
                  C        DIMENSION A(N),B(N)
          0004              DIMENSION A(22),B(22)
          0005             REAL*8 VDOT,A,B
          0006             VDOT = 0.
          0007             DO 10 I=1,N
          0008      10     VDOT = VDOT+A(I)*B(I)
          0009             RETURN
          0010             END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	76	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	52	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000	R*8	VDOT

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
2-00000008	I*4	I	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	AP-0000000C@	I*4	N

ARRAYS

Address	Type	Name	Bytes	Dimensions
AP-00000004@	R*8	A	176	(22)
AP-00000008@	R*8	B	176	(22)

```
VDOT                                                    6-Apr-1981 15:06:03     VAX-11 FORTRAN V2.0-2              Page   2
                                                        1-Dec-1980 07:14:24     _DBA0:[D11R.GCP]VDOT.FOR;7

LABELS

    Address   Label

      **      10

Total Space Allocated = 136 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:           0.59 seconds
  Elapsed Time:       15.48 seconds
  Page Faults:        280
  Dynamic Memory:     160 pages
```

```
0001            FUNCTION VMAG(A,N)
        C*********************************************************************
        C       ABSOLUTE MAGNITUDE OF A(N) VECTOR
        C*********************************************************************
0002            INCLUDE 'DEBUG.COM'
00100   0003    *       COMMON /DEBUG/ IENTER,IDEBUG
00200           * C
00300           * C             USER CONTROLLED PARAMETERS TO VARY DEBUG PRINT LEVEL
00400           * C
00500           * C                     INTER     IF 1, PRINTS WHEN ENTERS MOST SUBROUTINES
00600           * C                     IDEBUG    0-10, HIGHER NUMBER MEANS MORE PRINT
00700           * C
        0004            REAL*8 VMAG,A,VDOT
        0005            VMAG = SQRT(VDOT(A,A,N))
        0006            RETURN
        0007            END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	38	PIC CON REL LCL SHR EXE RD NOWRT LONG
2 $LOCAL	24	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 DEBUG	8	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000	R*8	VMAG

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
AP-00000004@	R*8	A	3-00000004	I*4	IDEBUG	3-00000000	I*4	IENTER	AP-00000008@	I*4	N

FUNCTIONS AND SUBROUTINES REFERENCED

MTH$DSQRT VDOT

Total Space Allocated = 70 Bytes

COMMAND QUALIFIERS

```
  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,ONAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:              0.45 seconds
  Elapsed Time:          10.41 seconds
  Page Faults:           279
  Dynamic Memory:        160 pages
```

```
0001          SUBROUTINE WET(TIEO,TIE)
        C*************************************************************
        C
        C     SUBROUTINE WET CREATES THE TRANSFORMATION FROM EARTH
        C     FIXED TO INERTIAL COORDINATES
        C
        C     INPUT VARIABLES
        C             TIEO    = INITIAL TRANSFORMATION
        C
        C     OUTPUT VARIABLES
        C             TIE     = EARTH FIXED TO INERTIAL TRANSFORMATION
        C                             AT TIME T
        C
        C     PROGRAMMED BY JACK MYERS 16JUNE1980
        C                     EXT 4443
        C
        C*************************************************************
        C
0002          INCLUDE 'TIME.COM'
      * C
0003  *      COMMON /TIME/ TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TZERO
      *     .          ,TMEAS,TRACK,TIS,TISN,DTA,DATER,TPRINT,DTPRINT
0004  *      REAL*8 TIME,TNEXT,TSTOP,TIA,DEL,TIN,DTN,DATEO,TMEAS,TRACK,TIS,
      *     .       TISN,DTA,TZERO,DATER,TPRINT,DTPRINT
      * C
      * C            THESE ARE THE TIME REFERENCE FRAMES
      * C
      * C                 TIME      ATOMIC TIME SINCE INITIALIZATION (SEC)
      * C                 TNEXT     TIME FOR NEXT POSITION INTEGRATION (SEC)
      * C                 TSTOP     RUN TERMINATION TIME (SEC)
      * C                 TIA       ATTITUDE INTEGRATION TIME (SEC)
      * C                 D L          "          "     STEP SIZE (SEC)
      * C                 TIN       POSITION INTEGRATION TIME (SEC)
      * C                 DTN          "          "     STEP SIZE (SEC)
      * C                 DATEO     DATE OF FLIGHT EPOCH (JD)
      * C                 DATER     DATE OF 1950 EPOCH (JD)
      * C                 TZERO     START TIME IN SECS. SINCE DATEO
      * C                 TSLEW     TIME NEEDED TO SLEW AND ACQUIRE (SEC)
      * C                 TIS       REAL WORLD REFERENCE TIME (SEC)
      * C                 TISN      TIME FOR NEXT RW POSITION INTEGRATION (SEC)
      * C                 DTA       USUALLY + DEL BUT + TSLEW - TIA WHEN DEL
      * C                           TOO LARGE AT MEASUREMENT TIME
      * C                 TPRINT    TIME FOR PRINT (SEC)
      * C                 DTPRINT   INCREMENT ON TPRINT (SEC)
      * C
0005          DATA WE/7.2921152E-5/ !RAD/SEC
0006          DIMENSION A(3,3)
0007          REAL*8 TIE,TI O,WE,A,DJDS,TE
        C**************************************************************
        C     COMPUTE TIME CHANGE SINCE DATER=JANUARY 1 1950
        C**************************************************************
0008          DJDS=24.*3600.*(DATEO-DATER)
0009          TE=DJDS+TZERO+TIME
        C**************************************************************
        C     SET UP TRANSFORMATION FOR ROTATION SINCE DATER
        C**************************************************************
0010          DO 10 I=1,3
```

```
                  DO 10 J=1,3
   10             A(I,J)=0.
                  A(1,1)=COS(WE*TE)
                  A(1,2)=-SIN(W *TE)
                  A(2,1)=-A(1,2)
                  A(2,2)=A(1,1)
                  A(3,3)=1.
C**********************************************************************
C                 SET UP TOTAL TRANSFORMATION
C**********************************************************************
                  CALL MATAB(A,TIE0,TIE,3,3,3)
                  RETURN
                  END
```

PROGRAM SECTIONS

Name	Bytes	Attributes
0 $CODE	149	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 $PDATA	4	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 $LOCAL	132	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
3 TIME	136	PIC OVR REL GBL SHR NOEXE RD WRT LONG

ENTRY POINTS

Address	Type	Name
0-00000000		WET

VARIABLES

Address	Type	Name	Address	Type	Name	Address	Type	Name	Address	Type	Name
3-00000038	R*8	DATE0	3-00000070	R*8	DATER	3-00000020	R*8	DEL	2-00000050	R*8	DJDS
3-00000068	R*8	DTA	3-00000030	R*8	DTN	3-00000080	R*8	DTPRINT	2-00000060	I*4	I
2-00000064	I*4	J	2-00000058	R*8	TE	3-00000018	R*8	TIA	AP-00000008@	R*8	TIE
AP-00000004@	R*8	TIE0	3-00000000	R*8	TIME	3-00000028	R*8	TIN	3-00000058	R*8	TIS
3-00000060	R*8	TISN	3-00000048	R*8	TMEAS	3-00000008	R*8	TNEXT	3-00000078	R*8	TPRINT
3-00000050	R*8	TRACK	3-00000010	R*8	TSTOP	3-00000040	R*8	TZERO	2-00000048	R*8	WE

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*8	A	72	(3, 3)

```
LABELS

    Address   Label

      **      10

FUNCTIONS AND SUBROUTINES REFERENCED

MATAB          MTH$DCOS       MTH$DSIN

Total Space Allocated = 421 Bytes

COMMAND QUALIFIERS

  FORTRAN /LIST GCP,INDATA,MATAB,OUTDATA,RUNG,DNAV,EPHEM,TRUEA,SPRESS,OCCULT,GPERT,GCPSEQ,VISIBLE,GENENV,TREG,GYROUT,RATE,BMAT,CMA

  /CHECK=(NOBOUNDS,OVERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOMACHINE_CODE  /CONTINUATIONS=19

COMPILATION STATISTICS

  Run Time:            0.98 seconds
  Elapsed Time:        13.67 seconds
  Page Faults:         346
  Dynamic Memory:      160 pages
```

3.0 SIMULATION MODE 2

The second mode of GCPSIM operation provides for extraction of GCP's directly from LANDSAT imagery. The main portion of GCPSIM is identical to that discussed previously. The primary difference is that several additional programs are hosted on the PDP 11/70 and communicate with GCPSIM, which is hosted on the VAX 11/780, through a parallel link (Figure A-45).

Two primary programs exist. The first called LMEX allows a landmark data base to be constructed independently from GCPSIM. The other main program performs the scrolling of the imagery, extraction of image data, and correlations from the stored GCP. These two programs are discussed separately in the following sections.

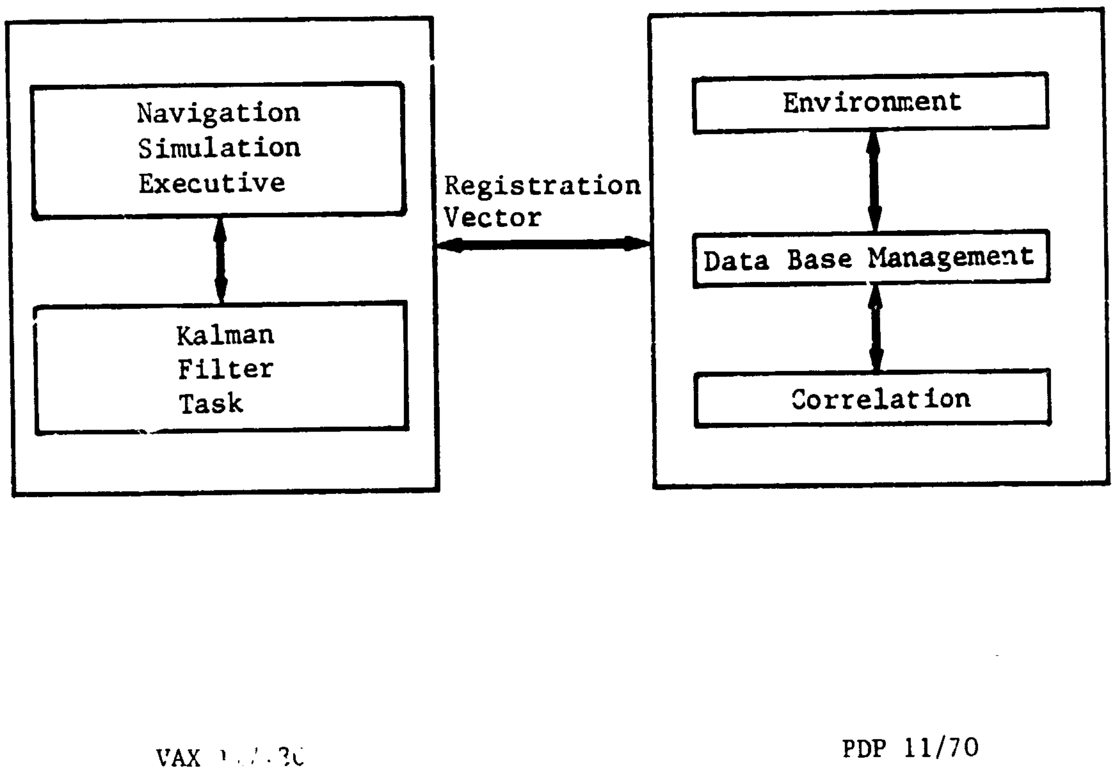


Figure A-45. Implementation Strategy of GCPSIM Landmark Extraction Mode

3.1 LANDMARK (GCP) EXTRACTION PROGRAM (LMEX)

The Landmark Extraction Program, LMEX, was written to allow the user to interactively extract ground control points from LANDSAT imagery. These extracted GCP's will be used for correlation purposes on later LANDSAT imagery of the same area.

LMEX created a data base that will be used with the correlation-scroll programs. LMEX can create a new data file or can be used to modify an existing data file.

The landmark data base file is a direct access file of record size 256 bytes. The user can name this file any name desired. The file set up is as follows:

Header block (4 records)
Landmark data (1 record per landmark) . . .

The first four records are header records containing information about each extracted landmark. Each landmark extracted uses up 16 bytes of header space, therefore up to 63 landmarks may be extracted. Word 1 of the header contains the number of landmarks in the data base. Each landmarks image data consists of 256 bytes. This allows each landmarks data to be stored in one record.

The LANDSAT imagery is displayed on a RAMTEK 9100 B/W graphics display system. The data displayed is either 7-bit or 6-bit grey scale imagery. The data being displayed will be scrolled from the bottom of the screen to the top, thus simulating the satellite motion.

The user may run LMEX on a previously stored landmark data base or create a new one. Landmarks may be defined at any time by using the joystick. The only restriction on landmark definition is that adjacent landmarks must be at least 200 lines apart. The LMEX programs checks for this occurring and will not allow it.

LMEX

Is this a new data base file to be used?

T: Initialize landmark no. & other variables as well as the header

F: Read in the landmark no. and header information

Get a line of image data and scroll it

Is a previously-defined landmark on the screen?

T: Draw a box around the l.m.

F: NULL

Does the user wish to define a landmark?

T:

Allow the user to position a cursor at the center of the landmark

Determine header offsets and other landmark information

Determine where in the header this landmark fits

Is this a valid landmark?

T: Store the landmark data & header

Draw a box around the landmark

F: NULL

F: NULL

Do until all the lines of imagery have been scrolled and the user wishes to exit

Write back the updated header block

Figure A-46

3.2 CORRELATION - SCROLL PROGRAMS

The correlation-scroll programs consist of three separate tasks. These tasks are installed and run concurrently. Inter-task communication is performed by using a shareable global area. Inter-task sychronization is performed by using global event flags. The three tasks are:

(1) SCRLL - This tasks scrolls the image data.

(2) SCOUNT - This tasks updates the landmark data base and decides when a correlation is to be performed.

(3) CORR - This task performs the actual corrleation and returns the registration vector.

These tasks run at different priorities. SCRLL is set at the highest priority, next is SCOUNT, and finally CORR. SCRLL will scroll a line of image data, signal for SCOUNT to become active, and then suspend itself until SCOUNT reawakes SCRLL. SCOUNT in turn, determines whether to awake CORR to perform landmark correlation.

The inter-relationship of the tasks may be seen in Figure A-46 .

CORRELATION - SCROLL PROGRAM STRUCTURE

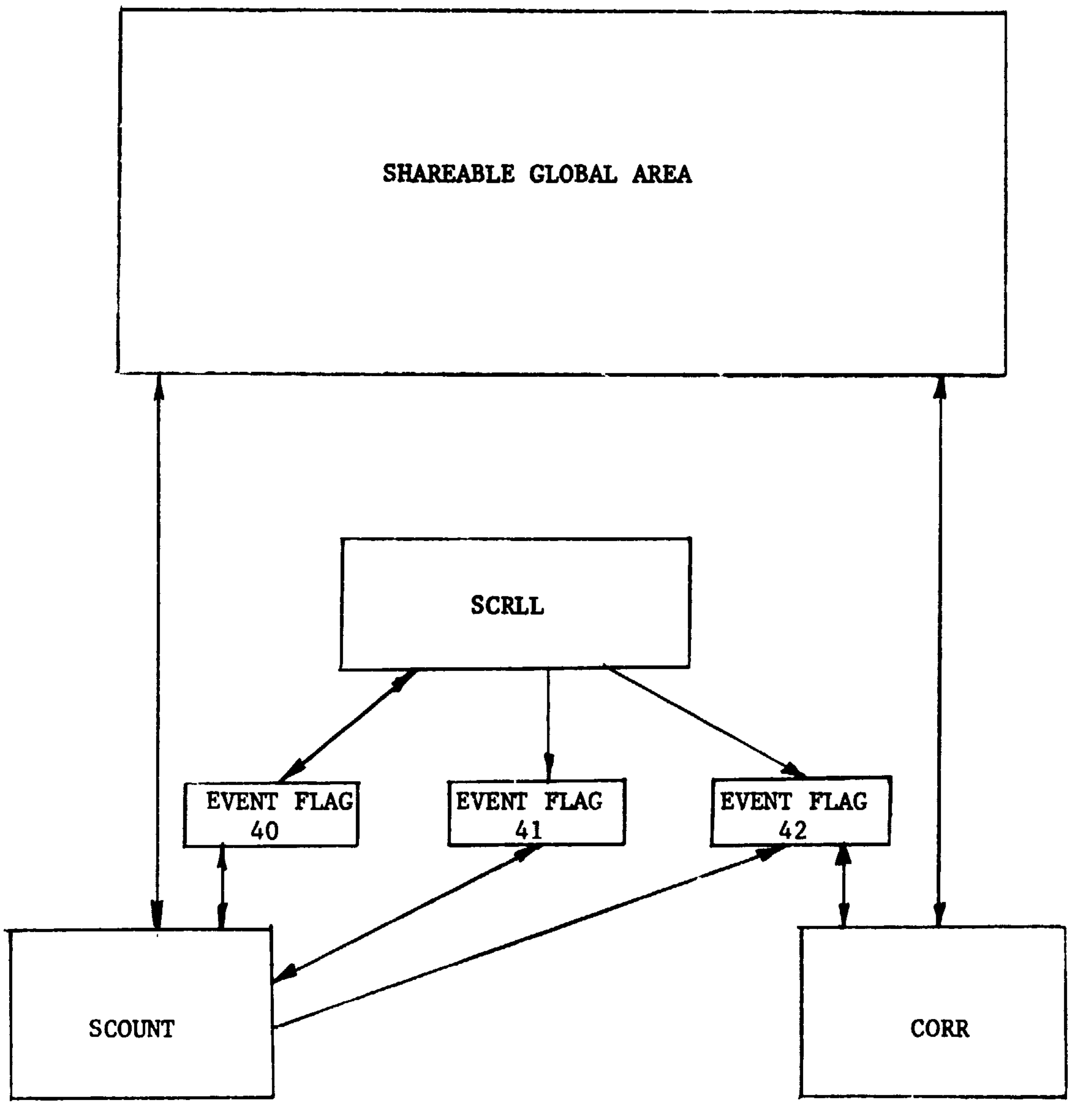


Figure A-47

3.2.1 SCROLL PROGRAM (SCRLL)

The SCRLL program is the driver task for the correlation-scroll programs. This program starts the other two tasks running. This program will then begin scrolling image data. After each line of image data is displayed and scrolled, SCOUNT will be activitated and SCRLL will suspend itself. SCRLL will then be activated by SCOUNT when SCOUNT has finished performing its function.

SCRLL

Reset The RAMTEK 9100

Set event flag 40. Reset flags 41 and 42.

Get the name of the image file to scroll & the starting element

Create a shareable global and an address window for mapping purposes

Start up SCOUNT and CORR to become active

Open the image data file. Set line counter.

- Get no. lines to be scrolled from GCPSIM

Do until end flag from GCPSIM

- Wait until event flag 40 is set
- Get a line of imagery and display it
- Scroll the image, simulating motion
- Clear event flag 40. Set event flag 41 to start SCOUNT processing.

Do while lines of imagery still need to be scrolled

Set the end flag and set event flags 41 and 42

Exit task off

Figure A-48

3.2.1.1 DRAW A BOX (DRAW)

DRAW performs a similar operation as the DRABOX library utility. DRAW should be used when the imagery that is to have the box drawn has been scrolled. The differences between DRAW and DRABOX have to do with the RM9100's window and scan parameters.

The passed auguments I and J define the upper left corner of the area to have the box enclose. The box is always 18 by 18 pixels, thus enclosing a 16 x 16 pixel landmark.

FORMAT: CALL DRAW (I-J)

<u>DRAW</u>

DRAW THE TOP LINE
DRAW THE BOTTOM LINE
DRAW THE LEFT LINE
DRAW THE RIGHT LINE

Figure A-49

3.2.2 SCROLL COUNTER (SCOUNT)

SCOUNT is the main data base handler for the correlation-scroll programs. This program keeps track of the number of lines that have been scrolled and whether or not a landmark has become visible. If a landmark is visible, this program waits unitl the entire search area is visible and then activites the landmark correlation program CORR.

SCOUNT is activated by SCRLL when event flag 41 is set. After each line of image data has been scrolled, SCOUNT will be activated to perform its checking and then deactivate itself and activate SCRLL to begin the process over.

SCOUNT

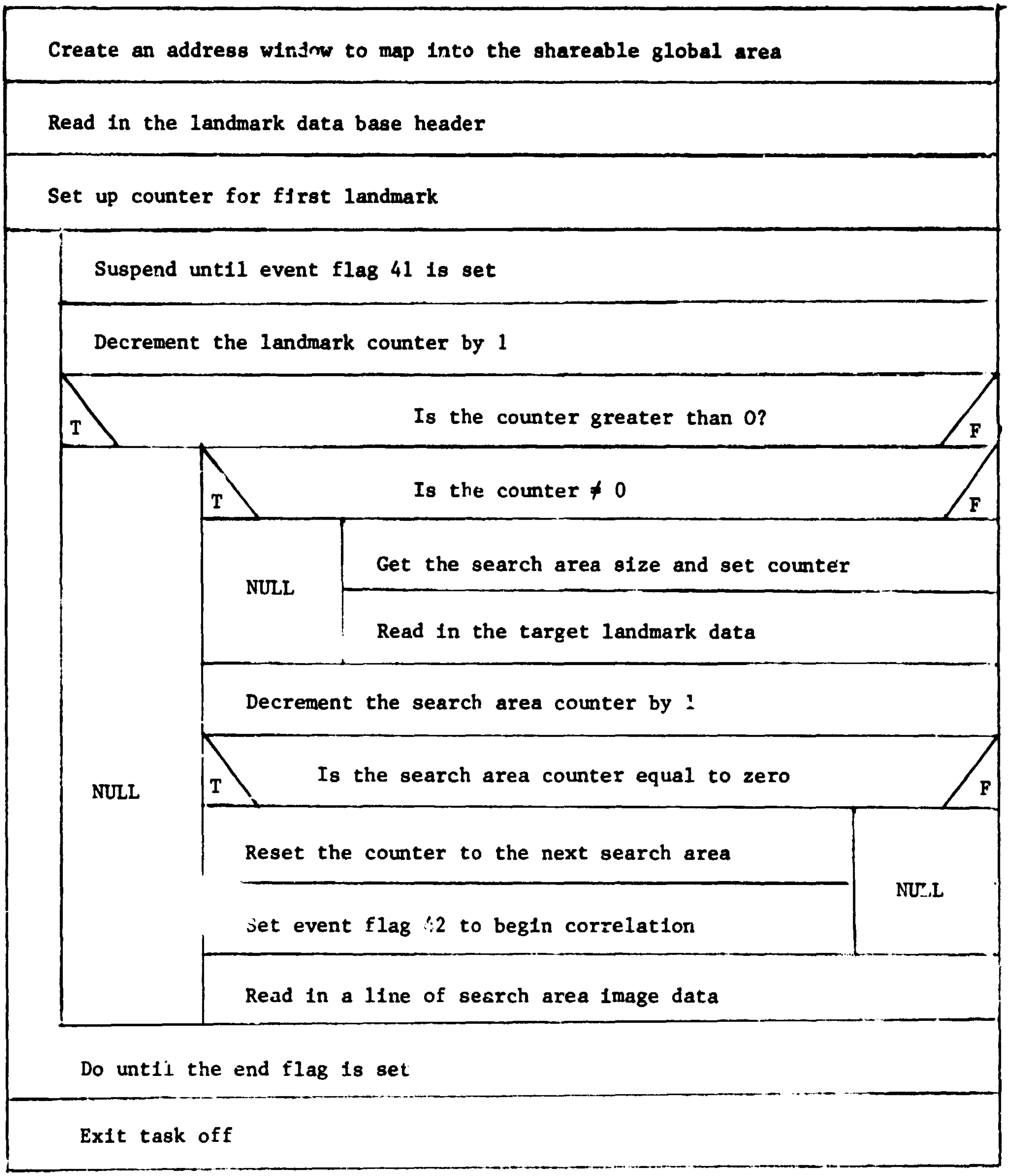


Figure A-50

3.2.3 CORRELATION (CORR)

CORR is used in conjunction with SCOUNT to perform the correlation analysis on the LANDSAT imagery. CORR returns the relative registration vector of the best fit of the lankmark imagery upon the target search area imagery.

CORR waits for event flag 42 to be set before any processing occurs. SCOUNT is the task which will set this flag. When SCOUNT sets this flag, the known landmark imagery has been stored in the array LM and the target search area image data is in the array WIN.

The SSDA algorithm uses an exhaustive search of the landmark data versus the search area data. The best fit is where the correlation value determined by the S.S.D.A algorighm is at a minimum. The S.S.D.A algorithm is:

$$\sum_{I=1}^{\#\text{ of lines}} \; \sum_{J=1}^{\#\text{ of elements}} \text{abs}\,(\text{WIN}(J,I)-\text{WINDO MEAN})-(\text{LM}(J,I)-\text{LANDMARK MEAN}))$$

Once CORR has determined the best fit landmark placement, the tasks suspends itself until SCOUNT starts it up again.

CORR

Create an address window to map into the shareable global area

- Suspend until event flag 42 is set
- Calculate the mean of the known landmark data
- Using the SSDA correlation, find the best fit location of the landmark on the target search area
- Draw a box around the best fit
- Reset event flag #42

Do until the end flag is set

Exit task off

Figure A-51

3.2.3.1 SSDA CORRELATION ROUTINE (CSDA)

CDSA implements the SSDA correlation algorithm. This routine exhaustively calculates the correlation values for a known landmark versus a target search area. The best fit is where the correlation value is at a minimum.

The algorithm is:

$$\sum_{I=1}^{\#\text{ of lines}} \sum_{J=1}^{\#\text{ of elements}} \text{abs}(\text{WIN}(J,I)-\text{WINDOW MEAN})-(\text{LM}(J,I)-\text{LANDMARK MEAN}))$$

FORMAT: CALL CSDA (WIN,LANMRK,WM,LM,N,BFE,BFL,M)

WIN	=	contains search area imagery
LANMRK	=	contains landmark area imagery
WM	=	search area mean
LM	=	landmark mean
N	=	landmark size in pixels
BFE	=	relative best bit element
BFL	=	relative best fit line
M	=	search area size

CSDA

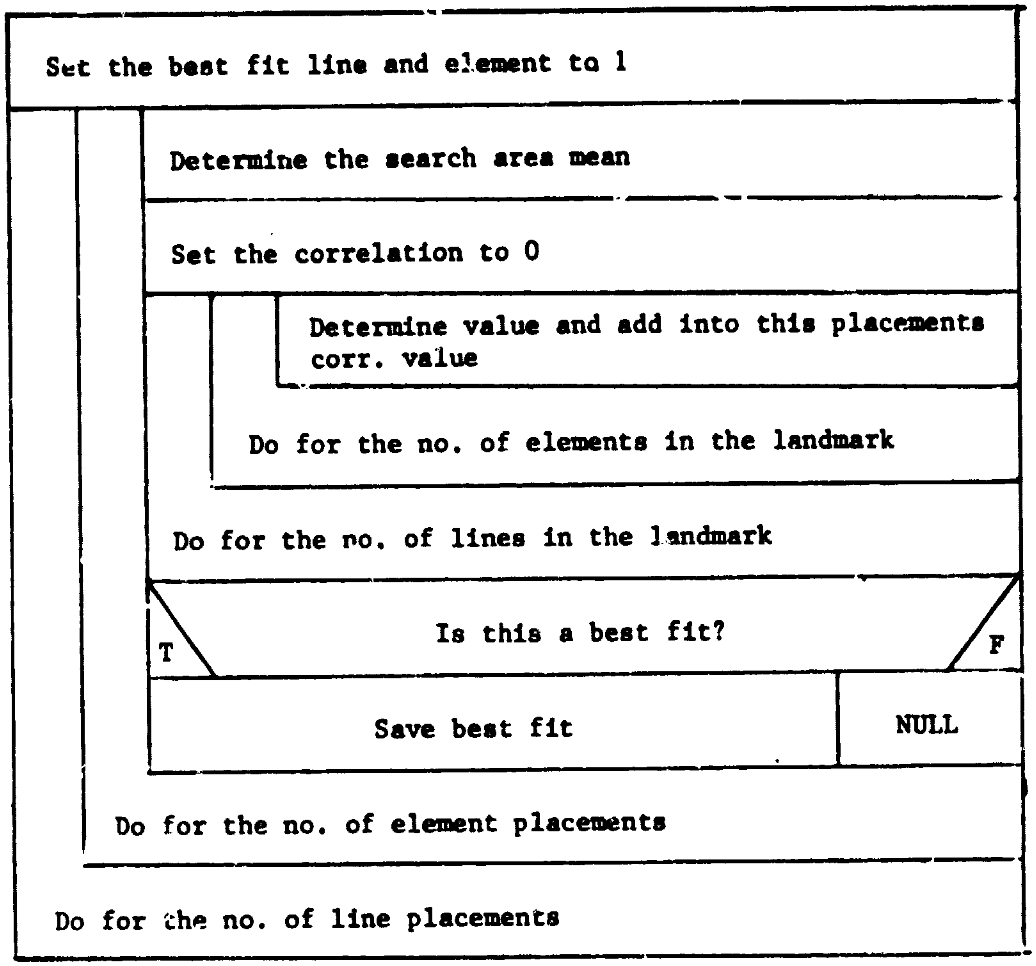


Figure A-52

3.3 LIBRARY ROUTINES

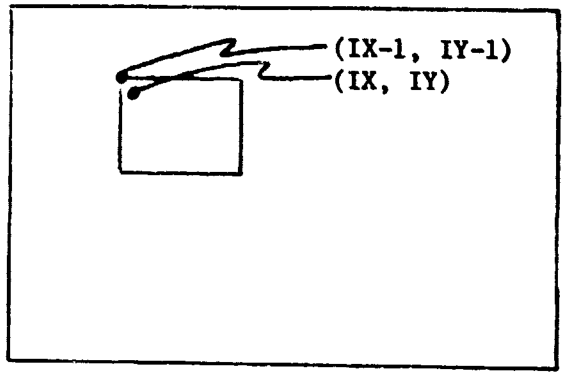
The library routines are those routines which perform a general function without being strictly I/O routines. These subroutines are:

1. DRABOX
2. MEAN
3. POS

C-5

3.3.1 DRAW A BOX (DRABOX)

The DRABOX routine draws a square box sized ISIZE x ISIZE. INMTENS defines the intensity that this box is to be written at on the RM9100. IX and IY define the upper left hand corner of the data to be enclosed by the box. IBUF is a scratch array.



FORMAT: CALL DRABOX (INTENS,ISIZE,IX,IY,IBUF)

Restrictions: $0 \leq ISIZE \leq 6$

<u>DRABOX</u>

Initialize the scratch array to the output intensity
Write the box on the RM9100.

Figure A-53

3.3.2 DETERMINE THE MEAN (MEAN)

The subroutine MEAN calculates the mean of a group of array elements. The array IARRAY (element, line) contains the data values. IARRAY is a square, 2 dimensional array. RWM is the calculated mean value. ISIZE is the number of elements and lines to be used. IES and ILS are the offsets into IARRAY from where the mean is to be taken.

FORMAT: CALL MEAN (IARRAY,RWMENA,ISIZE,ILS,IES)

MEAN

Initialize mean to zero

Sum up next element to previous sum

Do for the number of elements

Do for the number of lines

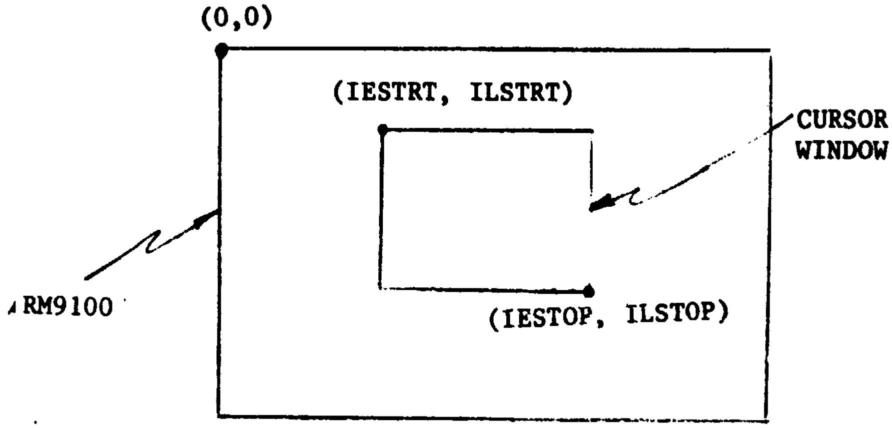
Divide the total sum by the number of elements x the number of lines

Figure A-54

3.3.3 POSITION CURSOR (POS)

POS is used to allow the user to position the cursor on the RM9100 screen. This routine allows a 9 x 9 pixel cursor to be drawn and moved around by the user. When switch 4 is flipped up, then down, the user has positioned the cursor to the desired location and control is returned to the calling routine.

IX and IY are the initial location of the center of the cursor. IX and IY also recieve the final values of the cursor center. ILSTRT, ILSTOP, IESTRT and IESTOP define a rectangular area in which the cursor may move.



FORMAT: CALL POS (IX,IY,ILSTRT,ILSTOP,IESTRT,IESTOP)

POS

T	Is the stick in detent?	F
NULL	Read the data under where the cursor is to be drawn and save	
	Draw the cursor	
	Redraw the data	
Do until switch 4 is up		

Figure A-55

3.4 I/O ROUTINES

The I/O routines are those routines which are used strictly for I/O purposes. These routines are:

	Subroutine Name	Section
1.	ADJDAT	
2.	INIJOY	
3.	RAMRDB	
4.	RAMWRB	
5.	READJY	
6.	RESET	
7.	RMCHCK	
8.	RMRDIM	
9.	RMSET	
10.	RMSETO	
11.	RMWRIM	
12.	WIO	

3.4.1 JOYSTICK I/O ROUTINES

The joystick device on the PDP 11/70 is terminal number TT4:. The joystick routines read this terminal and interpret the data. For data to be sent from TT4:, one of the switches marked 1 through 4 must be switched up.

3.4.1.1 ADJUST DATA (ADJUST)

The ADJUST subroutine is written in DEC MACRO assembly language. This routine adjusts the bits in the output data from the joystick. This routine is device dependent and called only by the subroutine READJY. The source code for ADJUST is found in the file ADJDAT.MAC.

3.4.1.2 INITIALIZE THE JOYSTICK (INIJOY)

The INIJOY subroutine assigns a logical unit number to the joystick. This subroutine is called once.

The parameter ILUN is the logical unit number to assign the joystick to. This unit number will be used by the routine READJY, to read the joystick device.

FORMAT: CALL INIJOY (ILUN)

3.4.1.3 READ THE JOYSTICK (READJY)

The READJY subroutine allows the user to read the joystick. Four words will be returned on each call. These words will be the switch word, X, Y, and Z words. The X, Y, and Z words have a range between 0 and 1023. When the stick is in detent the X and Y words are 512. When the stick is up in the upper left corner the X and Y words are 0. The Z word is set by the joystick knob. The switch word is bit oriented and follows this convention:

Bit Location	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													X	X	X	X
Switch Location													4	3	2	1

SWITCH WORD

If bit 0 is set switch 1 is up. If bit 1 is set, switch 2 is up, and so on. There are 4 switches.

The IWRDCT parameter must always be 4. ABUFF is the buffer to receive the 4 words of output. ILUN is the logical unit number of the joystick as initialized by INIJOY.

FORMAT: CALL READJY (ABUFF,IWRDCT,ILUN)

3.4.2 RAMTEK 9100 I/O ROUTINES

All RM9100 I/O routines use the WTQIO and GETADR system I/O subroutines. The basic structure for a RM9100 I/O routine is the following:

1. The command instruction is placed in a buffer.

2. The address of the command buffer and the length of the buffer is stored in a parameter array. This is where GETADR is used.

3. Depending on the function, the appropriate data from the parameter array is written/read to the RM9100. The operation of reading/writing is performed by using a WTQIO.

Two auxiliary routines that have been used by all RM9400 I/O subroutines are WIO and RMCHCK. These routines have been developed so that all I/O functions will utilize them and help conserve task space.

3.4.2.1 RM9100 READ IMAGE WINDOW (RAMRDB)

The RAMRDB subroutine allows the user to read a rectangular area from the RM9100 memory. This rectangular area is defined by the calling routine by setting IXSTRT, IXSTOP, IYSTRT and IYSTOP. Up to 2000 pixels may be read in any one call to RAMRDB. ISCAN defines the scan direction of the read. IDATA is the array in which data will be read and ILEN is the number of pixels to read into IDATA.

FORMAT: CALL RAMRDB (ISCAN,IXSTRT,IXSTOP,IYSTRT,IYSTOP,IDATA,ILEN)

3.4.2.2 RM9100 WRITE IMAGE WINDOW (RAMWRB)

The RAMWRB subroutine allows the user to write a rectangular area to the RM9100 memory. This rectangular area is defined by the calling routine by setting IXSTRT, IXSTOP, IYSTRT, and IYSTOP. Up to 2000 pixels may be written in any one call to RAMWRB. ISCAN is the scan direction of the write. IDATA is the array containing ILEN number of pixels to be written.

FORMAT: CALL RAMWRB (ISCAN, IXSTRT, IXSTOP, IYSTRT, IYSTOP, IDATA, ILEN)

3.4.2.3 RESET THE RM9100 (RESET)

The RESET instruction is an RM9100 system clear function identical to the power-on (or hard reset) initialization sequence. All pending operations are discarded. The refresh memory will be erased.

FORMAT: CALL RESET

3.4.2.4 CHECK RM9100 I/O (RMCHCK)

The RMCHCK subroutine checks to see if the previous RM9100 I/O was correctly handled by the RM9100. The status words are contained in the common block IO. The directive status word, IDS, is equal to 1 if the directive was sent successfully. The I/O status word, IOST(1), is 1 if the issued I/O executed correctly. If either of these variables do not equal 1, the I/O was unsuccessful.

FORMAT: CALL RMCHCK

3.4.2.5 RM9100 READ IMAGE (RMRDIM)

The RMRDIM subroutine allows the user to read a rectangular area from the RM9100 memory. Up to 2000 pixels may be read in any one call to RMRDIM. The default window of the entire screen is used.

ISCAN determines the primary and secondary scan directions.

ISCAN = 0	primary - left to right	secondary - top to bottom
ISCAN = 4	primary - top to bottom	secondary - left to right

IXS and IYS are the element and line values in memory at which the read is to begin. IDATA is the buffer in which data is to be filled and ILEN is the number of pixels to be read.

FORMAT: CALL RMRDIM (ISCAN,IXS,IYS,IDATA,ILEN)

3.4.2.6 SET UP RM9100 (RMSET)

The RMSET subroutine resets the window parameters of the RM9100 back to their original values. These are:

IXSTRT = 0

IXSTOP = 319

IYSTRT = 0

IYSTOP = 255

FORMAT: CALL RMSET

3.4.2.7 RESET RM9100 ORIGIN (RMSETO)

This routine allows the user to set the RM9100 origin parameters. Updating the y origin by one in a loop will give an illusion of vertical scrolling. This routine is used to scroll image data.

FORMAT: CALL RMSETO (IXORG,IYORG)

3.4.2.8 RM9100 WRITE IMAGE (RMWRIM)

The RMWRIM writes to any rectangular area in the RM9100, in word mode. Up to 2000 pixels may be written in any one call to RMWRIM. The default window of the entire screen is used.

In the LMEX programs, ICHNL always equals -1, ISCALE always equals 0, and IFCTN always equals 0. ISCAN is the scan direction to be used for the write.

ISCAN = 0	primary - left to right secondary - top to bottom
ISCAN = 4	primary - top to bottom secondary - left to right

IXS and IYS are the element, line start where the write to memory is the occur. IDATA is the buffer address of the element(s) to be written and ILEN is the number of pixels to be written.

FORMAT: CALL RMWRIM (ICHNL,ISCAN,ISCALE,IFCTN,IXS,IYS,IDATA,ILEN)

3.4.2.9 ISSUE WAIT FOR QUEUE I/O (WIO)

The WIO subroutine issues a PDP 11/70 WTQIO system directive. There are two arguments passed to this routine, IFCTN and IDEV. IFCTN specifies the I/O function to be executed and IDEV is the device in which the I/O function is to occur.

This subroutine is used for all RM9100 I/O. The common block IO contains the other parameters required by the WTQIO instruction.

FORMAT: CALL WIO (IFCTN,IDEV)

FUNCTION CODES: "410 - Write Logical block (RM9100)

"1010 - Read Logical block (RM9100)